REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (*Leave blank*)

2. REPORT DATE
Sep 94

3. REPORT TYPE AND DATES COVERED

4. TITLE AND SUBTITLE
Development of a Nonlinear Simulation for the McDonnell Douglas F-15 Eagle with a Longitudinal TECS Control-Law

5. FUNDING NUMBERS

6. AUTHOR(S)
James D. Dutton Jr.

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
AFIT Students Attending:
University of Washington

8. PERFORMING ORGANIZATION REPORT NUMBER
AFIT/CI/CIA
94-136

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)
DEPRTMENT OF THE AIR FORCE
AFIT/CI
2950 P STREET
WRIGHT-PATTERSON AFB OH 45433-7765

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for Public Release IAW 190-1
Distribution Unlimited
MICHAEL M. BRICKER, SMSgt, USAF
Chief Administration

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

19941207 074

14. SUBJECT TERMS

15. NUMBER OF PAGES
145

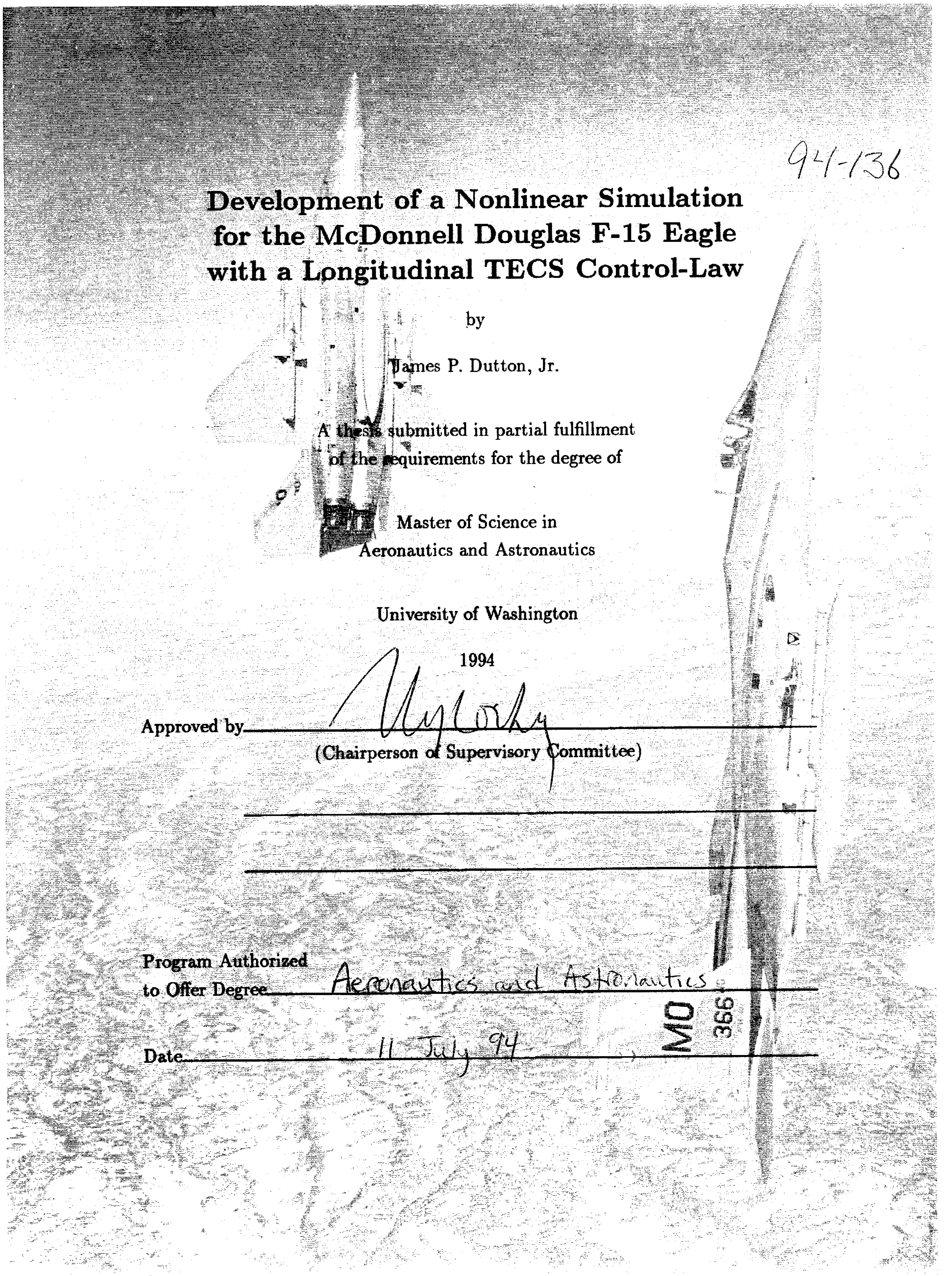
16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT

18. SECURITY CLASSIFICATION OF THIS PAGE

19. SECURITY CLASSIFICATION OF ABSTRACT

20. LIMITATION OF ABSTRACT



94-136

Development of a Nonlinear Simulation for the McDonnell Douglas F-15 Eagle with a Longitudinal TECS Control-Law

by

James P. Dutton, Jr.

A thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science in
Aeronautics and Astronautics

University of Washington

1994

Approved by______________________________
(Chairperson of Supervisory Committee)

____

____

Program Authorized
to Offer Degree Aeronautics and Astronautics

Date 11 July 94

Development of a Nonlinear Simulation for the McDonnell Douglas F-15 Eagle with a Longitudinal TECS Control-Law

by

James P. Dutton, Jr.

A thesis submitted in partial fulfillment
of the requirements for the degree of

Master of Science in
Aeronautics and Astronautics

University of Washington

1994

Approved by

(Chairperson of Supervisory Committee)

Program Authorized
to Offer Degree Aeronautics and Astronautics

Date 11 July 94

In presenting this thesis in partial fulfillment of the requirements for the Master of Science degree at the University of Washington, I agree that the Library shall make its copies freely available for inspection. I further agree that extensive copying of this thesis is allowable only for scholarly purposes, consistent with "fair use" as prescribed in the U.S. Copyright Law.

Signature James P [illegible] Jr.

Date 7/8/94

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

I would like to thank everyone who in one way or another has contributed to the completion of this thesis or in any other way has assisted along the path to completing this degree. In particular, thanks to the first controls crew—Dave, Joel, Marc, Clint, Pat, Cindy, and Janet—who put up with questions like "What's a partial fraction expansion again?" To the second crew—Zeek, Bruzzer, Mike & Phil, Tony, and Super Dave—for just plain making the program more fun. To Ewald, the only controls office guy who has remained since the beginning, thanks for your computer wizardry and LATEX mastery. Hope to see you in the Rhineland one day. And of course, endless thanks to Jane, who masterminded my acceptance into this program and who nearly threw a party when I turned in the last of many grad plan changes.

To Dr. Ly—I have had many very good advisors and instructors, but there is no doubt you have been one of the very best. Thanks for your extreme dedication to all your students, for the many hours you've sacrificed toward this thesis, and for your patience in dealing with yours truly.

I dedicate this thesis to Erin Elizabeth Ruhoff, who in less than four months will be my wife. We met in the first week of this Master's program, and she has uplifted and encouraged me every step of the way. Erin, we will be together forevermore. I love you.

Finally, and above all else, I give all honor and credit to our heavenly Father, who is the Source of it all.

Chapter 1

INTRODUCTION

1.1 Problem Description

The purpose of this report is to develop using SIMULINK an aircraft simulation that couples the nonlinear equations of motion with the nonlinear aerodynamic and engine performance data. The simulation has been designed using data from actual flight testing of the McDonnell Douglas F-15 Eagle. The ultimate objective is to provide a generic framework for the development of nonlinear simulations of other aircraft with minimal required changes. Design of a longitudinal autopilot using the Total Energy Control System (TECS) concept is accomplished using the linearized model and later validated with the nonlinear model.

1.2 Analysis of Data Provided

The following sections summarize the model characteristics and the relevant support modules for the F-15 described in [5]. The numerical data were provided in the format of a Genesis simulation used at Wright-Patterson and the coding was subsequently converted from FORTRAN to MATLAB. A detailed description of the model follows in Chapter 2.

The evaluation of the open-loop nonlinear model and the design of the TECS control law will be accomplished at the two flight points shown in Table 1.1. Other flight conditions throughout the flight envelope will also be investigated.

1.2.1 Model Characteristics

The model is an integration of several modules, each performing a specific function. These modules include the aerodynamic, propulsion, and atmospheric models (see Appendix E), as well as the nonlinear equations of motion. Modeling of the control

Table 1.1: *Selected Flight Conditions*

Flight Point	Altitude (ft)	V_{TAS} (ft/s)	Mach Number
1	9,800	539.1	0.5
2	30,000	497.3	0.5

surface and actuator dynamics as well as any sensor dynamics is performed where necessary in the SIMULINK environment. The closed-loop control law, in this case TECS, is also implemented in SIMULINK. Figure 1.1 summarizes the integration of these components to form the complete system model.

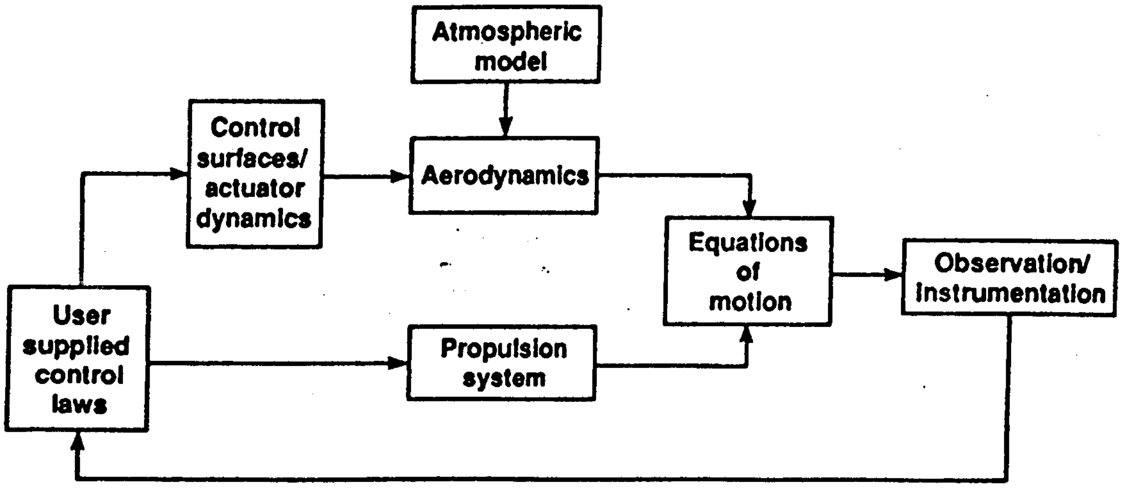

Atmospheric model
Control surfaces/ actuator dynamics
Aerodynamics
Equations of motion
Observation/ Instrumentation
User supplied control laws
Propulsion system

Figure 1.1: Integration of System Model Components.

The aircraft modeled is the McDonnell Douglas F-15 Eagle, the state-of-the-art in current-day operational, high-performance fighter aircraft. It is powered by two afterburning turbofan engines, each providing approximately 32,000 lb of thrust at maximum power. The primary flight control surfaces include horizontal stabilators capable of both symmetric and differential movement, conventional ailerons, and twin vertical rudders. There are a total of six actuators–two stabilators, two ailerons, and two rudders. All actuators are modeled identically with rate limits of 24 deg/sec and first-order response characteristics of

$$G(s) = \frac{20}{s + 20}$$

The individual surface position limits and sign conventions for positive deflection are summarized in Table 1.2. The aircraft mass and geometry characteristics are summarized in Table 1.3.

Table 1.2: *Flight Control Surfaces Characteristics*

Control Surface	Symbol	Limits	Sign Convention (+)
Symmetric stabilator	δ_H	±20°	Trailing edge down
Differential stabilator	δ_D	$+15°/-25°$	Left trailing edge down
Aileron	δ_A	±20°	Left trailing edge down
Rudder	δ_R	±30°	Trailing edge left

Table 1.3: *F-15 Mass and Geometry Characteristics*

Parameter	Symbol	Units	Value
Wing area	S	ft^2	608.0
Wing span	b	ft	42.8
Mean aerodynamic chord	c	ft	15.95
Aircraft weight	W	lb	45,000.0
Moments of inertia	I_x	$\text{slug}-\text{ft}^2$	28,700.0
	I_y	$\text{slug}-\text{ft}^2$	165,100.0
	I_z	$\text{slug}-\text{ft}^2$	187,900.0
Products of inertia	I_{xz}	$\text{slug}-\text{ft}^2$	-520.0
	I_{xy}	$\text{slug}-\text{ft}^2$	0.0
	I_{yz}	$\text{slug}-\text{ft}^2$	0.0

1.2.2 Aerodynamic Model

The aircraft aerodynamics are modeled using a combination of multidimensional tables and linear interpolation to form nonlinear function generators. The highly nonlinear aerodynamics encountered in the extreme portions of the aircraft flight envelope are therefore represented, which will be valuable in validating the control law after it has met the requirements for the linearized model. Most of the aerodynamic quantities are a function of Mach number M, and some combination of angle of attack α, sideslip angle β, and symmetric stabilator deflection δ_H.

The aerodynamic model calculates the nondimensional force and moment coefficients, which are then used to calculate the total associated forces and moments. The equations used for the coefficients are

- Coefficients of forces

$$\begin{aligned}
C_L &= C_{L_o} + \Delta C_{L_{n_z}} n_z \\
C_D &= C_{D_o} + \Delta C_{D_{alt}} + \Delta C_{D_{noz}} \\
C_Y &= C_{Y_o} + C_{Y_{\delta_A}} \delta_A + C_{Y_{\delta_D}} \delta_D - \Delta C_{Y_{\delta_R}} K_{\delta_{R_y}}
\end{aligned}$$

- Coefficients of moments

$$\begin{aligned}
C_\ell &= C_{\ell_o} + C_{\ell_{\delta_A}} \delta_A + C_{\ell_{\delta_D}} \delta_D - \Delta C_{\ell_{\delta_R}} K_{\delta_{R_\ell}} + \tfrac{b}{2V}(C_{\ell_p} p + C_{\ell_r} r) \\
C_m &= C_{m_o} + \Delta C_{m_{n_z}} n_z + \tfrac{\bar{c}}{2V}(C_{m_q} q + C_{m_{\dot\alpha}} \dot\alpha + C_{L_o} \Delta N_o) \\
C_n &= C_{n_o} + C_{n_{\delta_A}} \delta_A + C_{n_{\delta_D}} \delta_D + \Delta C_{n_{\delta_R}} K_{\delta_{Rn}} + \tfrac{b}{2V}(C_{n_p} p + C_{n_r} r)
\end{aligned}$$

The terms C, ΔC, ΔN, and K are outputs from the function generation routines, and are either calculated directly or by linear interpolation of the tabular data. In the case of the F-15, all coefficients are determined from tables except $\Delta C_{D_{noz}}$ and C_D for $\alpha > 40°$, which are calculated directly. The parameters affecting each of the coefficients are summarized in Table 1.4. The total forces and moments are calculated

Table 1.4: *Independent Parameters Affecting the Aerodynamic Coefficients*

Aero Coefficient	Independent Parameters	Aero Coefficient	Independent Parameters
C_{L_o}	M, α, δ_H	C_{ℓ_p}	M, α
$\Delta C_{L_{n_z}}$	M	C_{ℓ_r}	M, α
C_{m_o}	M, α, δ_H	C_{n_o}	M, α, β
$\Delta C_{m_{n_z}}$	M	$C_{n_{\delta_A}}$	M, α
C_{m_q}	M, α	$C_{n_{\delta_D}}$	M, α
$C_{m_{\dot{\alpha}}}$	M, α	$\Delta C_{n_{\delta_R}}$	M, α, β
ΔN_o	M	$K_{\delta_{Rn}}$	M, α
C_D		$K_{\delta_{R_\ell}}$	M
$(\alpha < 32)$	C_{L_o}, M	C_{n_p}	M, α
$(32 < \alpha < 40)$	C_{L_o}, M, α	C_{n_r}	M, α
$(\alpha > 40)$	C_{L_o}, α	C_{y_o}	M, α, β
$\Delta C_{D_{alt}}$	h	$C_{y_{\delta_A}}$	M, α
$\Delta C_{D_{noz}}$	M, δ_{PLA}	$C_{y_{\delta_D}}$	M, α
C_{ℓ_o}	M, α, β	$\Delta C_{y_{\delta_R}}$	M, α, δ_R
$\Delta C_{\ell_{\delta_R}}$	M, α, δ_R	$K_{\delta_{R_y}}$	M

from the equations

$$\begin{aligned} L &= \bar{q}SC_L \\ D &= \bar{q}SC_D \\ Y &= \bar{q}SC_Y \\ \\ \Sigma L &= \bar{q}SbC_\ell \\ \Sigma M &= \bar{q}S\bar{c}C_m \\ \Sigma N &= \bar{q}SbC_n \end{aligned}$$

where $\bar{q} = \frac{1}{2}\rho V^2$ is the dynamic pressure, S is the wing area, b is the wing span, and $\bar{c}$ is the mean aerodynamic chord.

1.2.3 Propulsion Model

The propulsion system model consists of two distinct engine models. Although the two engines are similar, they are not exactly identical—i.e. for a given throttle setting, the thrust produced may vary slightly. The engine thrust vectors are aligned with the aircraft x body-axis, and the thrust produced is a function of altitude h, Mach number M, and throttle setting δ_{PLA}. Each engine is modeled as a nonlinear system having two separate sections—a core engine and an afterburner (augmentor), each with its associated sequencing logic.

The throttle position inputs to the engine model are in degrees of power-level-angle (PLA), with a minimum position of 20° and a maximum of 127°. The core section responds to settings up to 83°, while the afterburner section begins to respond at a position of 91°. The core model has first-order dynamics and rate limiting to model spool-up effects, while the afterburner has a rate limiter and sequencing logic to model the fuel pump and pressure regulator effects.

1.2.4 Atmospheric Model

The atmospheric model's data is based on tables from the U.S. Standard Atmosphere (1962). This model calculates values for the speed of sound, the acceleration due to gravity, air density, viscosity, and ambient static pressure and temperature based on the aircraft altitude. Linear interpolation is used between table values for altitudes from 0 to 90 km.

1.2.5 Equations of Motion

The nonlinear equations of motion used in the system model are based on the derivation by Duke, Antoniewicz, and Krambeer in [6]. These equations model the six-degree-of-freedom dynamics of a rigid aircraft flying over a flat, nonrotating Earth. The derivations are outlined in Chapter 2.

Chapter 2

THE NONLINEAR F-15 MODEL

Linear aircraft models are useful in the early development of control-law design and in the analysis of vehicle dynamics over many flight conditions. However, because such models are only approximations of the aircraft behavior, it is valuable to the engineer to be able to verify the performance of a control law with the corresponding nonlinear model.

2.1 Derivation of Nonlinear State Equations

Most linearized aircraft models do not necessarily decouple into the longitudinal and lateral modes. However, if we include additional simplifying assumptions such as vehicle symmetry or a specific reference trajectory such as straight-and-level flight, these modes can be decoupled. Motion of an aircraft can be modeled by the responses of the rigid-body dynamics described by a set of six nonlinear simultaneous second-order differential equations. In the model to be developed here, these equations will be used without assuming any reference trajectory or vehicle symmetry. Instead, the model will assume a rigid aircraft of constant mass flying over a flat, nonrotating earth.

2.1.1 Reference Systems

The primary reference systems are the body-, the wind-, and the vehicle-carried, vertical-axis systems. Each has its advantages for use with a particular set of equations or variables.

The rotational equations of motion are most easily referenced to the body-axis system. The body-axis rotational rates are measureable within the aircraft by sensors fixed in the body frame. The body-axis system has its origin at the aircraft center of gravity, with the x-axis directed out the aircraft nose, the y-axis out the right wing, and the z-axis out the bottom of the aircraft. The positive direction for the body-axis rates (p, q and r), velocities (u, v and w), and moments (L, M and N) are shown in Figure 2.1.

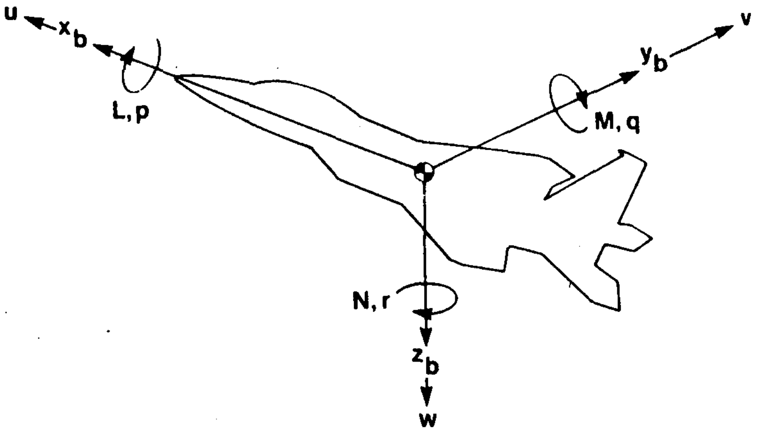


Figure 2.1: Vehicle Body-Axis System.

The wind-axis system is used for the translational equations of motion. Because aerodynamic forces act in the direction of the wind axes, such quantities as the angle of attack α, the total velocity V, and the sideslip angle β are directly measurable, or closely related to directly measureable quantities, in the aircraft. The x-axis in the wind-axis system is aligned with the aircraft velocity vector, with the y and z axes exiting the right side and the bottom of the aircraft respectively. Because both the wind- and body-axis systems have their origin at the center of gravity, their orientation can be defined from β and α. Components of the total velocity V can be expressed in terms of the body-axis velocities as,

$$\left\{ \begin{array}{rcl} u & = & V \cos\alpha \cos\beta \\ v & = & V \sin\beta \\ w & = & V \sin\alpha \cos\beta \end{array} \right. \qquad (2.1)$$

The vehicle-carried, vertical-axis system also has its origin at the aircraft center of gravity. This reference system is primarily useful for its orientation to the body-axis system, which is defined by the Euler angles ψ, θ and ϕ (Figure 2.2). The vehicle-carried, vertical-axis system is defined by having the x-axis directed north, the y-axis directed east, and the z-axis directed down. This axis system is simply the earth-fixed reference system translated to the aircraft center of gravity.

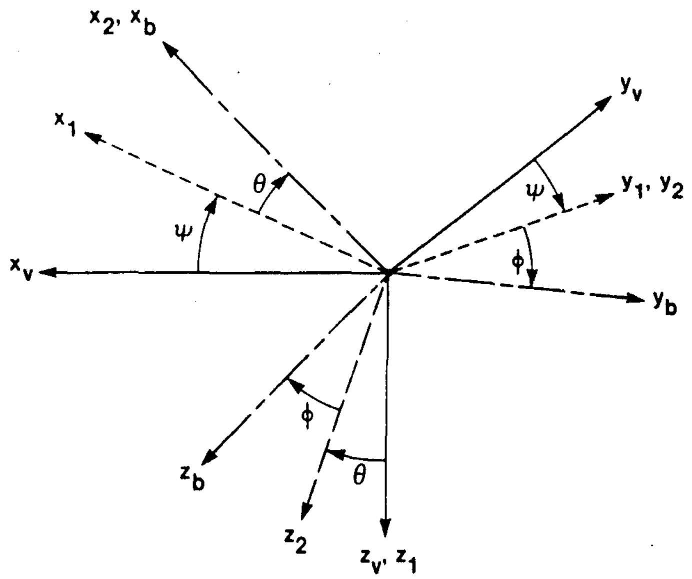


Figure 2.2: Orientation of Vehicle-Carried, Vertical-Axis System to the Body-Axis System.

Based on these axis systems, the aircraft dynamics can be described by 12 states, which will be divided into four sets of three variables, representing the vehicle rotational velocity (p, q, and r), the vehicle translational velocity (V, α, and β), the vehicle attitude (ϕ, θ, and ψ), and the vehicle position (h, x, and y).

2.1.2 Rotational Accelerations

The rotational acceleration terms $\dot{p}$, $\dot{q}$ and $\dot{r}$ are derived from the moment equation

$$\mathbf{M} = \frac{d}{dt}\mathbf{H} \tag{2.2}$$

where **M** is the total moment on the aircraft and **H** is the total angular momentum of the aircraft. For a moving reference frame such as the body-axis system, the time derivative operator $\delta/\delta t$ is used in place of d/dt, and the total angular momentum is given by $\mathbf{H} = I\Omega$, where I is the inertia tensor and Ω is the rotational velocity vector. The result is

$$\mathbf{M} = \frac{\delta}{\delta t}(I\Omega) + \Omega \times (I\Omega) \tag{2.3}$$

where

$$\mathbf{M} = \begin{bmatrix} \Sigma L \\ \Sigma M \\ \Sigma N \end{bmatrix} = \begin{bmatrix} L & + & L_T \\ M & + & M_T \\ N & + & N_T \end{bmatrix} \tag{2.4}$$

with L, M and N defined as the aerodynamic total moments about the x, y and z body axes respectively, and L_T, M_T and N_T defined as the sums of all the thrust-induced moments. The inertia tensor I is defined as

$$I = \begin{bmatrix} I_x & -I_{xy} & -I_{xz} \\ -I_{xy} & I_y & -I_{yz} \\ -I_{xz} & -I_{yz} & I_z \end{bmatrix} \tag{2.5}$$

where the moments of inertia about the x, y and z body axes and the products of inertia in the x-y, x-z and y-z body-axis planes are

- Moments of inertia:

$$I_x = \int_B (y^2 + z^2)\, dm$$

$$I_y = \int_B (x^2 + z^2)\, dm$$

$$I_z = \int_B (x^2 + y^2)\, dm$$

- Products of inertia:

$$I_{xy} = \int_B xy\, dm$$

$$I_{xz} = \int_B xz\, dm$$

$$I_{yz} = \int_B yz\, dm$$

The term Ω is a vector of the rotational rates p, q and r about the x, y and z body axes respectively. Because constant mass is assumed for the aircraft, the inertia tensor is constant with respect to time, and equation (2.3) can be solved for the rotational accelerations,

$$\frac{\delta}{\delta t}\Omega = \begin{bmatrix} \dot{p} \\ \dot{q} \\ \dot{r} \end{bmatrix} = I^{-1}\left[\mathbf{M} - \Omega \times (I\Omega)\right] \tag{2.6}$$

In order to simplify the expansion of this expression, the inverse of the inertia tensor will be defined as

$$I^{-1} = \frac{1}{\det I}\begin{bmatrix} I_1 & I_2 & I_3 \\ I_2 & I_4 & I_5 \\ I_3 & I_5 & I_6 \end{bmatrix} \tag{2.7}$$

where

$$\begin{aligned} \det I &= I_x I_y I_z - I_x I_{yz}^2 - I_z I_{xz}^2 - 2 I_{yz} I_{xz} I_{xy} \\ I_1 &= I_y I_z - I_{yz}^2 \\ I_2 &= I_{xy} I_z + I_{yz} I_{xz} \\ I_3 &= I_{xy} I_{yz} + I_y I_{xz} \\ I_4 &= I_x I_z - I_{xz}^2 \\ I_5 &= I_x I_{yz} + I_{xy} I_{xz} \\ I_6 &= I_x I_y - I_{xy}^2 \end{aligned}$$

and the terms D_x, D_y and D_z are defined as

$$\begin{aligned} D_x &= I_z - I_y \\ D_y &= I_x - I_z \\ D_z &= I_y - I_x \end{aligned}$$

Expressions for the rotational accelerations in equation (2.6) can now be expanded into the following set of scalar equations,

$$\left\{\begin{aligned}
\dot{p} &= \tfrac{1}{\det I}[(\Sigma L)I_1 + (\Sigma M)I_2 + (\Sigma N)I_3 - p^2(I_{xz}I_2 - I_{xy}I_3)\\
&\quad +pq(I_{xz}I_1 - I_{yz}I_2 - D_zI_3) - pr(I_{xy}I_1 + D_yI_2 - I_{yz}I_3)\\
&\quad +q^2(I_{yz}I_1 - I_{xy}I_3) - qr(D_xI_1 - I_{xy}I_2 + I_{xz}I_3) - r^2(I_{yz}I_1 - I_{xz}I_2)]\\
\dot{q} &= \tfrac{1}{\det I}[(\Sigma L)I_2 + (\Sigma M)I_4 + (\Sigma N)I_5 - p^2(I_{xz}I_4 - I_{xy}I_5)\\
&\quad +pq(I_{xz}I_2 - I_{yz}I_4 - D_zI_5) - pr(I_{xy}I_2 + D_yI_4 - I_{yz}I_5)\\
&\quad +q^2(I_{yz}I_2 - I_{xy}I_5) - qr(D_xI_2 - I_{xy}I_4 + I_{xz}I_5) - r^2(I_{yz}I_2 - I_{xz}I_4)]\\
\dot{r} &= \tfrac{1}{\det I}[(\Sigma L)I_3 + (\Sigma M)I_5 + (\Sigma N)I_6 - p^2(I_{xz}I_5 - I_{xy}I_6)\\
&\quad +pq(I_{xz}I_3 - I_{yz}I_5 - D_zI_6) - pr(I_{xy}I_3 + D_yI_5 - I_{yz}I_6)\\
&\quad +q^2(I_{yz}I_3 - I_{xy}I_6) - qr(D_xI_3 - I_{xy}I_5 + I_{xz}I_6) - r^2(I_{yz}I_3 - I_{xz}I_5)]
\end{aligned}\right. \tag{2.8}$$

2.1.3 Translational Accelerations

Derivation of the translational acceleration terms is based on the force equation

$$\mathbf{F} = \frac{d}{dt}(m\mathbf{V}) \tag{2.9}$$

where **F** represents the total force acting on the aircraft, m is the aircraft mass, and **V** is the total velocity. By assuming m to be constant and using the time derivative operator for a rotating reference frame $\delta/\delta t$, this expression can be expanded to

$$\mathbf{F} = m\,(\frac{\delta}{\delta t}\mathbf{V} + \Omega \times \mathbf{V}) \tag{2.10}$$

where $\mathbf{F} = [\Sigma X \;\; \Sigma Y \;\; \Sigma Z]^T$, a vector containing the sums of the aerodynamic, thrust, and gravitational forces in the x, y, and z body axes respectively, and $\mathbf{V} = [u \;\; v \;\; w]^T$. With some manipulation, equation (2.10) can be solved for the body-axis translational accelerations,

$$\frac{\delta}{\delta t}\mathbf{V} = \begin{bmatrix} \dot{u} \\ \dot{v} \\ \dot{w} \end{bmatrix} = \frac{1}{m}\mathbf{F} - \Omega \times \mathbf{V} \tag{2.11}$$

However, as previously mentioned, the desired form for the translational accelerations is in the wind-axis system—in terms of the total velocity magnitude V, the angle of attack α, and the sideslip angle β. The relationships between the translational accelerations of the body- and wind-axis systems are shown in equations (2.1). Derivation of the scalar equations for the wind-axis translational accelerations is involved and will not be shown here. The resulting equations are

$$\left\{\begin{array}{rcl} \dot{V} & = & \frac{1}{m}[-D\cos\beta + Y\sin\beta + X_T\cos\alpha\cos\beta + Y_T\sin\beta + Z_T\sin\alpha\cos\beta \\ & & \quad -mg(\cos\alpha\cos\beta\sin\theta - \sin\beta\sin\phi\cos\theta - \sin\alpha\cos\beta\cos\phi\cos\theta)] \\ \dot{\alpha} & = & \frac{1}{Vm\cos\beta}[-L + Z_T\cos\alpha - X_T\sin\alpha + mg(\cos\alpha\cos\phi\cos\theta + \sin\alpha\sin\theta)] \\ & & \quad +q - \tan\beta(p\cos\alpha + r\sin\alpha) \\ \dot{\beta} & = & \frac{1}{mV}[D\sin\beta + Y\cos\beta - X_T\cos\alpha\sin\beta + Y_T\cos\beta - Z_T\sin\alpha\sin\beta \\ & & \quad +mg(\cos\alpha\sin\beta\sin\theta + \cos\beta\sin\phi\cos\theta - \sin\alpha\sin\beta\cos\phi\cos\theta)] \\ & & \quad +p\sin\alpha - r\cos\alpha \end{array}\right. \tag{2.12}$$

where D is the total aerodynamic drag, Y is the total aerodynamic side force, L is the total aerodynamic lift, and X_T, Y_T and Z_T are the total thrust forces in the x, y and z body-axis directions respectively.

2.1.4 Attitude Rates

The Euler angle rates in the earth-fixed axis system and the rotational velocities in the body-axis system are related by a transformation matrix T, where

$$T = \begin{bmatrix} 1 & 0 & -\sin\theta \\ 0 & \cos\phi & \sin\phi\cos\theta \\ 0 & -\sin\phi & \cos\phi\cos\theta \end{bmatrix} \tag{2.13}$$

The equation for the transformation from the earth-fixed system to the body-axis system is

$$\Omega = T\,(\frac{d}{dt}\mathbf{E}) \tag{2.14}$$

with $\mathbf{E}$ representing a vector of the Euler angles, $\mathbf{E} = [\phi \quad \theta \quad \psi]^T$. By a simple rearrangement, equations for the attitude rates are

$$\frac{d}{dt}\mathbf{E} = T^{-1}\Omega \tag{2.15}$$

which can be expanded into the following scalar equations

$$\begin{cases} \dot{\phi} &= p + q\sin\phi\tan\theta + r\cos\phi\tan\theta \\ \dot{\theta} &= q\cos\phi - r\sin\phi \\ \dot{\psi} &= q\sin\phi\sec\theta + r\cos\phi\sec\theta \end{cases} \tag{2.16}$$

2.1.5 Earth-Relative Velocity

The earth-relative velocities are related to the body-axis velocities by

$$\mathbf{V} = L_{BV}\,(\frac{d}{dt}\mathbf{R}) \tag{2.17}$$

where $\mathbf{R}$ is the location of the aircraft in the earth-axis system, $\mathbf{R} = [x \quad y \quad z]^T$, and $z = -h$. Note that $\mathbf{V}$ is the total velocity vector with components u, v and w in the body-axis system. The transformation from the earth-axis system to the body-axis system is accomplished by the matrix L_{BV}, where

$$L_{BV} = \begin{bmatrix} \cos\psi & -\sin\psi & 0 \\ \sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi & \cos\phi \end{bmatrix} \tag{2.18}$$

Equation (2.17) is easily solved for the earth-relative velocities in terms of the body-axis velocities:

$$\frac{d}{dt}\mathbf{R} = L_{BV}^{-1}\,\mathbf{V} \tag{2.19}$$

Using this equation and equation (2.1), the earth-relative velocities can be solved in terms of V, α and β. The resulting equations are

$$
\left\{
\begin{aligned}
\dot{h} &= V(\cos\alpha\cos\beta\sin\theta - \sin\beta\sin\phi\cos\theta - \sin\alpha\cos\beta\cos\phi\cos\theta) \\
\dot{x} &= V[\cos\alpha\cos\beta\cos\theta\cos\psi + \sin\beta(\sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi) \\
&\qquad + \sin\alpha\cos\beta(\cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi)] \\
\dot{y} &= V[\cos\alpha\cos\beta\cos\theta\sin\psi + \sin\beta(\cos\phi\cos\psi + \sin\phi\sin\theta\sin\psi) \\
&\qquad + \sin\alpha\cos\beta(\cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi)]
\end{aligned}
\right. \tag{2.20}
$$

2.2 Nonlinear Simulation Model

The twelve nonlinear simultaneous second-order differential equations derived in Section 2.1 are implemented in the simulation in the form of a MATLAB special function, or *s-function.* The intricacies of this function are left to the reader to learn; however, the basic format and flow will be described here. Integration with the simulation program SIMULINK will also be discussed in general terms to give the reader an overall picture of how the various components of the simulation interact.

2.2.1 Component Integration—the S-Function

Systems saved under SIMULINK as s-functions behave like MATLAB m-functions and can be called from user-written routines or from the command line. The s-function makes it possible for the user to write customized routines for simulation, linearization, and parameter estimation. As described in Chapter 1, the F-15 dynamics are contained in two primary routines, `f25aero` and `f25eng`, quantifying the aircraft aerodynamic and propulsion characteristics respectively. The routine `atmos` calculates the current atmospheric parameters such as density and temperature to be used in these routines. Using the updated values of the system states and inputs, the necessary aerodynamic and propulsion parameters are calculated, which are then used in the calculation of the twelve state derivatives. All three of these routines were provided in FORTRAN format from the AIAA Design Challenge [5]. The s-function used for the open-loop simulation is shown in Appendix A.

Initial conditions for the aircraft states are also specified in the s-function. These values are of particular importance when determining the aircraft trim point for a given flight condition. This will be discussed in further detail in Section 2.3.2. The s-function also allows the system output values to be specified. For an open-loop model, the outputs selected are arbitrary. In this case, the twelve states are selected in order to validate the equilibrium state responses at the two flight conditions.

2.2.2 The Simulink Model

While the s-function allows us to specify a particular set of ordinary differential equations, the SIMULINK environment allows the user to design the overall system including other more generic operations such as inputs, filters, and feedback loops.

The open-loop SIMULINK system is shown in Figure 2.3. The primary use for this simulation format is to establish a trim point at selected flight points and evaluate the open-loop aircraft responses to initial conditions, control inputs, or disturbances. The initial conditions are specified in the appropriate s-function, here called `f25sfn` for the open-loop model. Additional inputs can be set for the aircraft. These include the aircraft flight-control surfaces, which consist of horizontal stabilators capable of symmetric (*DH*) and differential (*DD*) movement, conventional ailerons (*DA*), and twin vertical rudders (*DR*). In this report, these terms will be referred to as δ_H, δ_D, δ_A, and δ_R respectively. Thrust from the two engines is specified by the right and left power-level-angles, *PLAR* and *PLAL* respectively, or can be set identically as *PLASYM*. Again, the nomenclature in this text for these variables is δ_{PLAR}, δ_{PLAL} and δ_{PLA}. The outputs include the twelve states, plus the angle-of-attack rate $\dot{\alpha}$.

2.3 Linearization of the Model

Design of a feedback control system for a nonlinear system is greatly simplified by the use of a linearized model about some nominal trajectory. For aircraft, it is common practice to use several models linearized about various points within the flight envelope to design the control system. Normally, the controller structure is designed to be constant throughout the flight envelope, with some method of gain scheduling used to achieve adequate performance at all points. The linearized model is therefore a significant tool in simplifying the development of a satisfactory controller for the real-world, nonlinear aircraft.

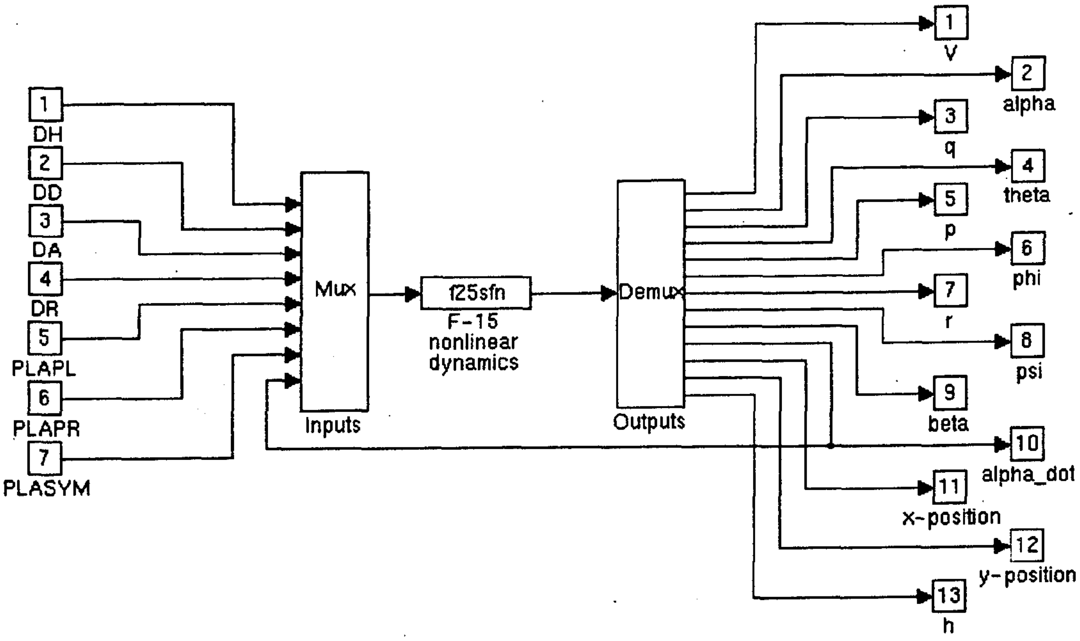


Figure 2.3: SIMULINK Model for Open-Loop F-15 Simulation.

2.3.1 Mathematical Approach

In this section, the linearization of the aircraft equations of motion will be summarized. The reader is referred to [6] for a more detailed derivation. Beginning with the translational equations of motion (2.12), by allowing the thrust and aerodynamic forces to be represented as the total external forces X, Y and Z, the result is

$$\left\{\begin{array}{rcl} \dot{V} & = & -g(\sin\theta\cos\alpha\cos\beta - \cos\theta\sin\phi\sin\beta - \cos\theta\cos\phi\sin\alpha\cos\beta) \\ & & +\frac{X}{m}\cos\alpha\cos\beta + \frac{Y}{m}\sin\beta + \frac{Z}{m}\sin\alpha\cos\beta \\ \dot{\alpha} & = & q - p\cos\alpha\tan\beta - r\sin\alpha\tan\beta + g\frac{\cos\theta\cos\phi\cos\alpha + \sin\theta\sin\alpha}{V\cos\beta} \\ & & -\frac{X}{m}\frac{\sin\alpha}{V\cos\beta} + \frac{Z}{m}\frac{\cos\alpha}{V\cos\beta} \\ \dot{\beta} & = & p\sin\alpha - r\cos\alpha + \frac{g}{V}(\sin\theta\cos\alpha\sin\beta + \cos\theta\sin\phi\cos\beta \\ & & -\cos\theta\cos\phi\sin\alpha\sin\beta) - \frac{X}{mV}\cos\alpha\sin\beta + \frac{Y}{mV}\cos\beta - \frac{Z}{mV}\sin\alpha\sin\beta \end{array}\right. \tag{2.21}$$

The linearized equations are now determined in terms of the perturbation variables ΔV, $\Delta\alpha$, $\Delta\beta$, Δp, Δq, Δr, ΔX, ΔY, ΔZ, $\Delta\theta$ and $\Delta\phi$ in a symmetric climb with

$$V = V_o + \Delta V \tag{2.22}$$

where V_o is the aircraft trim velocity (a constant). Similar relationships as in equation (2.22) exist for all the other variables listed above. In this example, for a symmetric climb, several trim values are equal to zero; namely, $\beta_o = p_o = q_o = r_o = Y_o = \phi_o = \psi_o = 0$. These trim values may not be zero for other flight conditions, such as a level turn. Because ΔV, $\Delta\alpha$, $\Delta\beta$, Δp, Δq, Δr, ΔX, ΔY, ΔZ, $\Delta\theta$ and $\Delta\phi$ are perturbations about their respective trim values, they are always treated as small quantities.

By substituting the expanded variables of the form of equation (2.22) into equations (2.21), linearized equations of motion for the perturbed variables ΔV, $\Delta\alpha$ and $\Delta\beta$ are derived. This is accomplished by assuming all second- and higher-order terms to be negligible (i.e. $\Delta V\,\Delta\alpha \approx 0$) and using the small angle approximations, $\cos\Delta\alpha \approx 1$ and $\sin\Delta\alpha \approx \Delta\alpha$. Leaving some relatively complicated manipulations to the reader, the equations become

$$\left\{\begin{array}{lcl} \Delta\dot{V} & = & -g\cos(\theta_o-\alpha_o)\,\Delta\theta + \frac{\cos\alpha_o}{m}\,\Delta X + \frac{\sin\alpha_o}{m}\,\Delta Z \\ \Delta\dot{\alpha} & = & \Delta q - \frac{g}{V_o}\sin(\theta_o-\alpha_o)\,\Delta\theta - \frac{\sin\alpha_o}{mV_o}\,\Delta X + \frac{\cos\alpha_o}{mV_o}\,\Delta Z \\ \Delta\dot{\beta} & = & \sin\alpha_o\,\Delta p - \cos\alpha_o\,\Delta r + \frac{g}{V_o}\cos\theta_o\,\Delta\phi + \frac{1}{mV_o}\,\Delta Y \end{array}\right. \tag{2.23}$$

Further, by expressing the external forces X and Z again in terms of the aircraft lift L, drag D, and thrust T and using the perturbation variables as described in equation (2.22), the linearized equations of translational accelerations are reduced to

$$\left\{\begin{array}{lcl} \Delta\dot{V} & = & -g\cos(\theta_o-\alpha_o)\,\Delta\theta - \frac{L_o}{m}\,\Delta\alpha - \frac{1}{m}\,\Delta D + \frac{\cos(\alpha_o+\alpha_T)}{m}\,\Delta T \\ \Delta\dot{\alpha} & = & \Delta q - \frac{g}{V_o}\sin(\theta_o-\alpha_o)\,\Delta\theta - \frac{D_o}{mV_o}\,\Delta\alpha - \frac{1}{mV_o}\,\Delta L - \frac{\sin(\alpha_o+\alpha_T)}{mV_o}\,\Delta T \\ \Delta\dot{\beta} & = & \sin\alpha_o\,\Delta p - \cos\alpha_o\,\Delta r + \frac{g}{V_o}\cos\theta_o\,\Delta\phi + \frac{1}{mV_o}\,\Delta Y \end{array}\right. \tag{2.24}$$

The linearized equations for rotational accelerations are found in a similar manner, defining the rotational velocities (p, q and r) and the moments (ΣL, ΣM and ΣN) in terms of their trim and perturbation values. Using the assumed level climb condition, we have $p_o = q_o = r_o = 0$ and $\Sigma L_o = \Sigma M_o = \Sigma N_o = 0$. Substituting the trim and perturbed variables into equations (2.8) and retaining only the first-order terms in Δp, Δq and Δr, the following linearized equations result,

$$I_{yy}\,\Delta\dot{q} - I_{xy}\,\Delta\dot{p} - I_{yz}\,\Delta\dot{r} = \Delta(\Sigma M) \tag{2.25}$$

$$I_{xx}\,\Delta\dot{p} - I_{xz}\,\Delta\dot{r} = \Delta(\Sigma L) \tag{2.26}$$

$$I_{zz}\,\Delta\dot{r} - I_{zx}\,\Delta\dot{p} = \Delta(\Sigma N) \tag{2.27}$$

By assuming a vehicle geometry that is symmetric about the body x-z plane, we have $I_{xy} = I_{yz} = 0$ and equation (2.25) can be further reduced to

$$I_{yy}\,\Delta\dot{q} = \Delta(\Sigma M) \tag{2.28}$$

Finally, the linearized attitude (Euler) rate equations are determined by substituting into equations (2.16) the appropriate trim and perturbation variables. After applying small angle approximations and eliminating higher-order terms, the resulting linearized equations are

$$\left\{\begin{array}{lcl} \Delta\dot{\theta} & = & \Delta q \\ \Delta\dot{\psi} & = & \frac{1}{\cos\theta_o}\Delta r \\ \Delta\dot{\phi} & = & \Delta p + \tan\theta_o \Delta r \end{array}\right. \tag{2.29}$$

2.3.2 Determination of the Trim Point

When a system is nonlinear, an operating point must be chosen at which to extract the linearized model. For the F-15 or any other aircraft, the equilibrium (or trim) values for the system states and inputs are determined for a given flight condition. In this case, the MATLAB algorithm `trim` is used. This routine proves adequate in providing the required data, however, several drawbacks exist which suggest the need for an improved program in future research.

The purpose of the `trim` routine is to determine steady-state parameters that satisfy input, output, and state conditions. This is accomplished by searching for the inputs $\mathbf{u}$ and states $\mathbf{x}$ that set the state derivatives to zero. Initial starting guesses

$\mathbf{x_o}$, $\mathbf{u_o}$ and $\mathbf{y_o}$ are given to the algorithm, where $\mathbf{y_o}$ are any outputs which are not also states. The routine `trim` then attempts to minimize the difference between the final steady-state values and these initial guesses, while driving the state derivatives to zero.

In this case, because the equilibrium condition is steady, level and unaccelerated flight, the trim states for q, p, ϕ, r, ψ and β can be assumed to be zero. Level flight is assured by setting the flight-path angle γ to be zero. Subsequent values for the elevator δ_H, throttle δ_{PLA}, aileron δ_A, and rudder δ_R and for the remaining states V, α and θ should exist for a steady-state solution. The `trim` routine, however, does not allow the user to set any of the states or outputs to a prespecified constant value, while then searching the remaining states, inputs, and outputs for the equilibrium values. Instead, `trim` considers such values as "desired" and iterates about all states until equilibrium or some close proximity to equilibrium is found.

As a result, in this nonlinear model the states that are known to be zero all take on finite, though small values. The flight-path angle also is nonzero for the equilibrium values selected by `trim`. An example of the disparity between the desired and the selected values is shown for FP1 (flight point one) in Table 2.1. The initial guesses come from the trim values generated by the Genesis model at Wright-Patterson AFB. Although the disparities between Genesis and `trim` are small enough to be sufficient for the purpose of deriving a linear model, the primary drawback of this technique is the added time required for `trim` to search all the states, instead of just those which are known to be nonzero.

2.4 Using the Nonlinear Model

This section will give a brief overview of the current capabilities of the F-15 nonlinear simulation model as well as the essential modifications needed for use with other aircraft.

Table 2.1: *Trim Data FP1 (9,800 ft, 0.5 M)*

Parameter	Units	Initial Guess	Trim Value	State Derivative
V	ft/s	539.08	539.08	-7.1636^{-9}
α	rad	0.0801	0.0801	-1.8594^{-9}
q	rad/s	0	-2.79^{-19}	-2.1328^{-5}
θ	rad	0.0801	0.0803	-2.7949^{-19}
p	rad/s	0	-2.54^{-20}	2.3895^{-19}
ϕ	rad	0	-1.02^{-18}	-2.8053^{-20}
r	rad/s	0	-3.22^{-20}	-1.8936^{-20}
ψ	rad	0	-1.66^{-13}	-3.2388^{-20}
β	rad	0	-8.41^{-21}	-2.9103^{-20}
δ_h	deg	-2.827	-2.880	N/A
δ_{PLA}	deg	37.4	37.4	N/A
δ_a	deg	0	0	N/A
δ_r	deg	0	0	N/A
γ	rad	0	0.0002	N/A

2.4.1 Applications

The nonlinear F-15 simulation model is comprised essentially of the governing s-function, which describes the interconnection of the key components which include the nonlinear equations of motion, and the supporting atmospheric, aerodynamic, and propulsion models. Because of the ease of implementing an s-function in the SIMULINK environment, the model can be used for both the open- and closed-loop evaluation. The procedure involves creating an overall system in SIMULINK in which the F-15 dynamics are described by a single s-function block. For example, referring back to Figure 2.3, the block titled "`F-15 nonlinear dynamics`" contains the open-loop s-function `f25sfn`. The MATLAB m-file for `f25sfn` is found in Appendix A.

It is likely that for other applications, the user may desire to change the inputs, outputs, or even the states of the original model set-up. As an example, such a change was made for its use in conjunction with the TECS control law. The changes to these parameters are made in the s-function, as can be seen by comparing the open-loop s-function `f25sfn` with the closed-loop TECS s-function `f25sfncl` in Appendix A. A sample of these changes is shown below for clarification.

- Open-loop s-function `f25sfn` input and output listings:

```
%%% INPUTS (U) %%%
DH          = u(1);          %%% SYMETRIC STABILATOR (DEG)      %%%
DD          = u(2);          %%% DIFFERENTIAL STABILATOR (DEG) %%%
DA          = u(3);          %%% AILERON DEFLECTION (DEG)       %%%
DR          = u(4);          %%% RUDDER DEFLECTION (DEG)        %%%
PLAPL       = u(5);          %%% LEFT  PLA (DEG)                %%%
PLAPR       = u(6);          %%% RIGHT PLA (DEG)                %%%
PLASYM      = u(7);          %%% SYMMETRIC PLA (DEG)            %%%
alpdot      = u(8);          %%% AOA RATE (RAD/S)               %%%

%%% SYSTEM OUTPUTS (Y) %%%
sys(1,1) = x(1);                                 %%% V          %%%
sys(2,1) = x(2);                                 %%% ALPHA      %%%
sys(3,1) = x(3);                                 %%% Q          %%%
sys(4,1) = x(4);                               %%% THETA      %%%
sys(5,1) = x(5);                               %%% P          %%%
sys(6,1) = x(6);                               %%% PHI        %%%
sys(7,1) = x(7);                               %%% R          %%%
sys(8,1) = x(8);                               %%% PSI        %%%
sys(9,1) = x(9);                               %%% BETA       %%%
sys(10,1) = al3+(al1+al2)/(V*m*COSBETA);       %%% ALPHA_DOT  %%%
sys(11,1) = x(10);                             %%% X-POSITION %%%
sys(12,1) = x(11);                             %%% Y-POSITION %%%
sys(13,1) = x(12);                             %%% H          %%%
```

- Closed-loop s-function `f25sfncl` input and output listings:

```
%%% INPUTS (U) %%%
DH         = u(1)+utrim(1);      %%% SYMETRIC STABILATOR (DEG) %%%
PLAPL      = u(2)+utrim(2);      %%% LEFT  PLA (DEG)           %%%
PLAPR      = u(3)+utrim(3);      %%% RIGHT PLA (DEG)           %%%
FLAPS      = u(4)+utrim(4);      %%% FLAPS (DEG)               %%%
DA         = u(5)+utrim(5);      %%% AILERON DEFLECTION (DEG)  %%%
DR         = u(6)+utrim(6);      %%% RUDDER  DEFLECTION (DEG)  %%%

%%% SYSTEM OUTPUTS (Y) %%%
sys(1,1)    = (V1+V2+V3+V4+V5)/m/g;                 %%% V_DOT/G  %%%
sys(2,1)    = x(1)-xx(1);                           %%% V        %%%
sys(3,1)    = x(12)-xx(12);                         %%% H        %%%
sys(4,1)    = x(4)-xx(4)-x(2)+xx(2);                %%% GAMMA    %%%
sys(5,1)    = x(3)-xx(3);                           %%% Q        %%%
sys(6,1)    = x(4)-xx(4);                           %%% THETA    %%%
sys(7,1)    = x(2)-xx(2);                           %%% ALPHA    %%%
sys(8,1)    = x(5)-xx(5);                           %%% P        %%%
sys(9,1)    = x(6)-xx(6);                           %%% PHI      %%%
sys(10,1)   = be6+(be1+be2+be3+be4+be5)/(m*V);      %%% BETA_DOT %%%
sys(11,1) = x(9)-xx(9);                             %%% BETA     %%%
sys(12,1) = x(8)-xx(8);                             %%% PSI      %%%
sys(13,1) = r*COSPHI*SECTHETA+q*SINPHI*SECTHETA; %%% PSI_DOT  %%%
sys(14,1) = x(7)-xx(7);                          %%% R        %%%
```

Note that the closed-loop inputs declaration includes the addition of the trim input initial conditions and the output declaration includes the subtraction of the respective trim initial condition, if one exists for that parameter. This is required for the closed-loop s-function since the values feeding into the controller (system outputs) and the values of the controls taken out of the controller (system inputs) are *perturbations.*

If the user desires to run the model simulation with a particular set of initial conditions, then in most cases this is best done by declaring these parameters as $\mathbf{x_o}$ and $\mathbf{u_o}$ vectors in the s-function. This is particularly appropriate whenever other blocks in the overall SIMULINK system contain integrators of first or higher order, such as in the case of a shaping filter. An excerpt from an s-function where the initial conditions are listed internally is:

```
function [sys,x0,u0] = f25sfncl(t,x,u,flag);
if flag == 0
  %%% SYSTEM CHARACTERISTICS/INITIAL CONDITIONS %%%
      sys = [12 0 13 8 12 0];
      %%% Trim Flight Point 2 %%%
      x0 = [497.31  0.18534 0.00000 0.18944 0.00000 ...
            0.00000 0.00000 0.00000 0.00000 0 0 30000];
      u0 = [-6.9712 0 0 0 76.750 80.679 0 0];

                          •
                          •
                          •
```

However, in some instances such as open-loop simulation, it may be convenient to set these values in the MATLAB memory and then include them in the call statement for the simulation. An example MATLAB listing is:

```
%%% Trim Flight Point 2 %%%
x0 = [497.31  0.18534 0.00000 0.18944 0.00000 ...
```

```
     0.00000 0.00000 0.00000 0.00000 0 0 30000];
%%% Call Simulation %%%
[t,x,y] = euler('f25sim',x0);
```

Note that in the above case, the initial conditions for the controls must be included in the s-function as shown in the first example or as a value in a step-input block in the SIMULINK system. The function `euler` identifies the method of numerical integration the simulation will use, and `f25sim` is the overall SIMULINK system within which the s-function is a single block.

2.4.2 Modification for Other Aircraft

Because of the modular design of this nonlinear simulation model, modification for other aircraft is made relatively easy—the user need only supply the modified aerodynamic and the propulsion models. The proper format for these modules is found from the examples shown in Appendix D. A summary of the required parameters to be returned from each module is shown below.

- Aerodynamic module:

C_L	Coefficient of Lift	ΣL	Total Rolling Moment
C_D	Coefficient of Drag	ΣM	Total Pitching Moment
C_Y	Coefficient of Side Force	ΣN	Total Yawing Moment

- Propulsion module:

X_T	Thrust in X Body-Axis Direction
Y_T	Thrust in Y Body-Axis Direction
Z_T	Thrust in Z Body-Axis Direction

The nonlinear equations of motion do not change regardless of the aircraft, so the actual s-function should require only slight modifications. It is even possible to use linearized aerodynamic and propulsion models while still utilizing the same basic equations of motion for the nonlinear vehicle dynamics.

Many of the system parameters, including the aircraft dimensions, the products of inertia, and the moments of inertia, are defined in the vector **A**, which is read into the aerodynamic, propulsion, and atmospheric modules of the simulation. This vector is also used to store updated values for the various atmospheric, aerodynamic, and propulsion parameters as well as the states and control inputs. The **A**-vector originally came from the Genesis simulation and contained 2,000 parameters. However, most of the parameters are either not used or are specific to the Genesis simulation. A listing of the significant **A**-vector parameters is shown in Appendix E. A sample of commands for running both the open- and closed-loop simulations in MATLAB is shown in Appendix F.

Chapter 3

MODEL EVALUATION AND LINEARIZATION

In this section, the nonlinear F-15 model will be analyzed by evaluating its open-loop characteristics in a steady unaccelerated level flight condition. Aircraft responses based on the initial states derived from the MATLAB `trim` function will be compared with those previously generated by the Genesis simulation at Wright-Patterson. Some of the challenges with the model will also be discussed, namely the limitations found in the model's accuracy throughout the flight envelope and the difficulties encountered in determining a trim point at a specified flight condition.

The model analysis will also include an evaluation of the linearized models at two flight points. State responses to a pulse-input independently applied to the elevator, rudder, and aileron control will be compared with those of the nonlinear model.

3.1 Open-Loop Nonlinear Model at Trim

Open-loop nonlinear simulation is used to validate the selected trim point for each of the flight conditions. Once validated, the trim points will be used to construct the linearized, state-space models to be used in the TECS control-law design.

3.1.1 Comparison to Genesis Simulation

In determining the trim points for the two flight conditions, the "initial guess" used for the aircraft trim states was obtained from the Genesis model. Values obtained from the MATLAB `trim` function differ slightly from those obtained by the Genesis simulation, as shown in Table 3.1.

Results of these variations in terms of aircraft responses are shown in Figures 3.1-3.2, with the complete set of plots for both flight points shown in Appendix B. In the case of the first flight condition (9800 ft, 0.5 M), the MATLAB initial conditions result in only very small deviations from the equilibrium. The largest of these is the altitude h, with an increase of 15 feet over a time interval of 100 seconds, primarily due to a nonzero flight path angle ($\alpha \neq \theta$, Table 3.1). Responses from the Genesis initial conditions show significant oscillations and a large deviation from the equilibrium. Since the oscillations dampen with time, this response is considered to be stable.

The second flight condition (30,000 ft, 0.5 M) shows larger deviations in the responses with the MATLAB initial conditions. The pitch angle θ decreases approximately 0.1° and h increases approximately 180 feet over a period of 100 seconds. Both the velocity V and the pitch rate q also show small deviations from the equilibrium values. With the Genesis initial conditions, larger deviations from trim are observed, with the exception of the altitude variable. The Genesis results produce oscillations with a significantly higher frequency than those obtained from the MATLAB trim condition.

Table 3.1: *Comparison of Trim Conditions*

States/ Outputs	Units	FP 1 MATLAB	FP 1 Genesis	FP 2 MATLAB	FP 2 Genesis
V	ft/s	539.08	539.08	497.32	497.32
α	rad	0.0801	0.0801	0.1853	0.1891
q	rad/s	0.0	0.0	0.0	0.0
θ	rad	0.0803	0.0801	0.1898	0.1891
p	rad/s	0.0	0.0	0.0	0.0
ϕ	rad	0.0	0.0	0.0	0.0
r	rad/s	0.0	0.0	0.0	0.0
ψ	rad	0.0	0.0	0.2197	0.0
β	rad	0.0	0.0	0.0	0.0
δ_h	deg	-2.827	-2.880	-6.970	-7.059
δ_{PLA}	deg	37.39	37.40	78.83	78.95
δ_a	deg	0.0	0.0	0.0	0.0
δ_r	deg	0.0	0.0	0.0	0.0

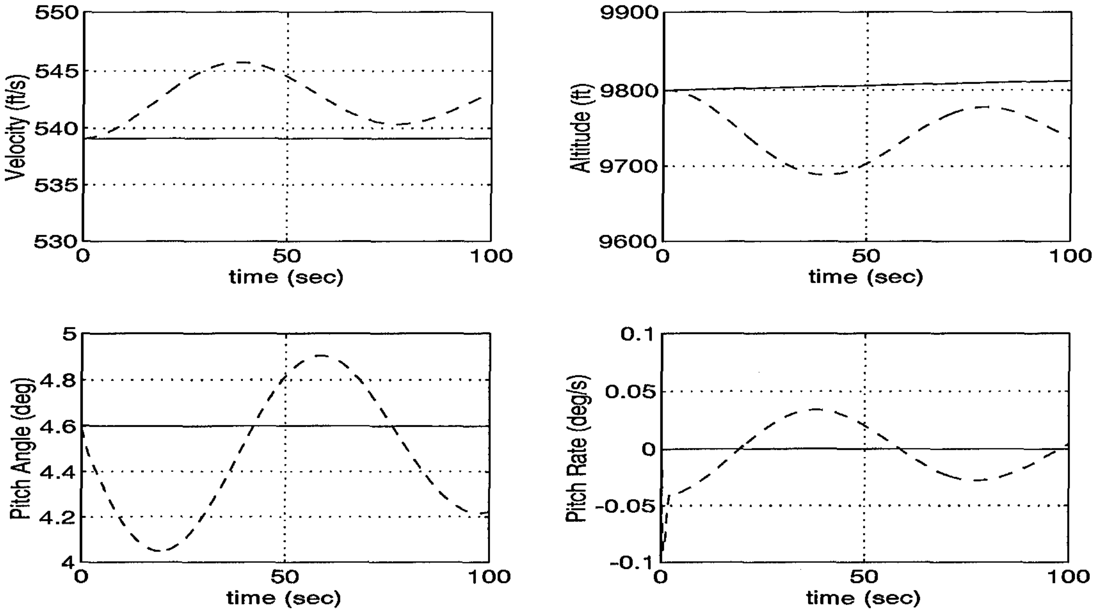


Figure 3.1: Flight Point 1—Comparison of Aircraft Responses to Initial Conditions Set at Trim Values: MATLAB (solid line) and Genesis (dashed line).

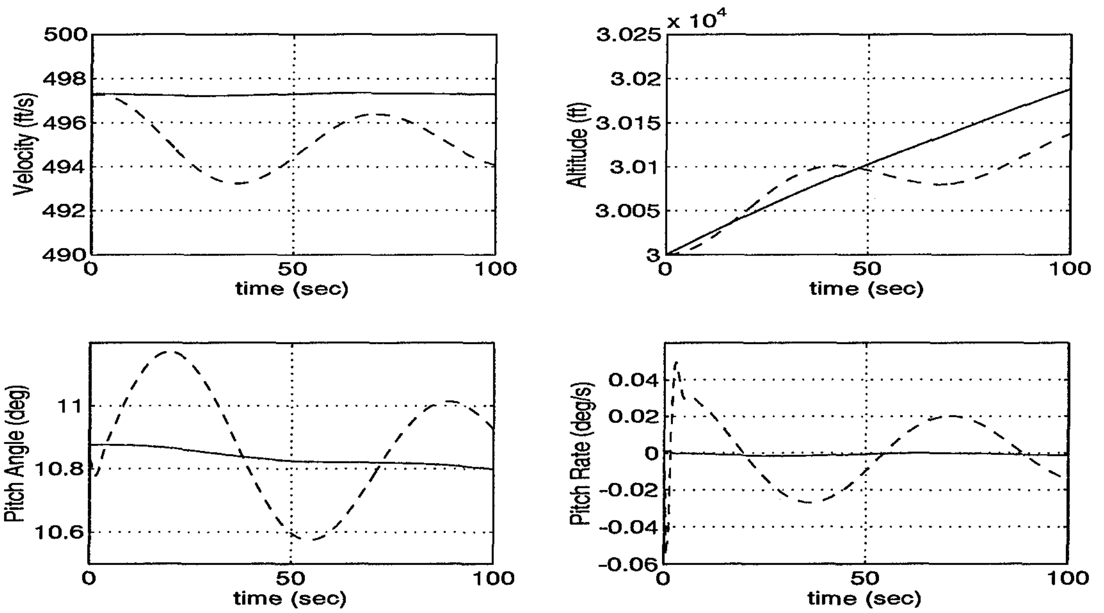


Figure 3.2: Flight Point 2—Comparison of Aircraft Responses to Initial Conditions Set at Trim Values: MATLAB (solid line) and Genesis (dashed line).

3.1.2 Limitations in the Flight Envelope

The two flight points selected for the evaluation of the nonlinear model are at the same Mach number (0.5 M). Although it is desireable to evaluate the model over as much of the flight envelope as possible, the higher Mach number regime created problems for the nonlinear model. As seen in Figure 3.3, at an altitude of 9,800 feet and Mach number of 0.9 (V = 970.34 ft/s), high-frequency oscillations take place after approximately 30 seconds of simulation. It is possible that these oscillations are the results of inaccurate aerodynamic table data. Because the purpose of the model is to provide a generic framework for many other aircraft, problems specific to the F-15 nonlinear modeling will be left for future investigation.

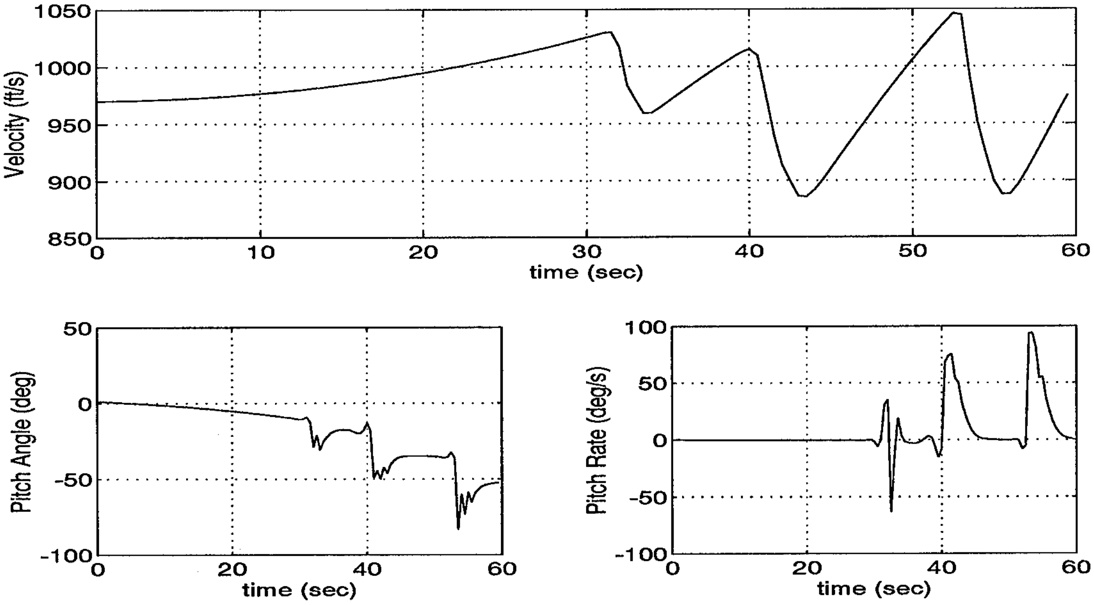


Figure 3.3: High-Frequency Oscillations at Higher Mach Numbers (9,800 ft, 0.9 M)

3.1.3 Limitations in Determining the Trim Point

Although the challenges associated with the MATLAB `trim` function were already addressed in the previous chapter, an additional problem became evident when evaluating the aircraft equilibrium responses. As mentioned in Section 3.1.1, the `trim` routine was run using initial guesses based on the trim values from the Genesis routine.

Clearly, this method would not be of practical use to other aircraft models, since such accurate estimates of the trim conditions are usually not available.

In order to test the reliability of the `trim` routine, starting values for the control settings were taken to be different from the original Genesis values, while keeping the same values for the velocity V, the angle of attack α, and the pitch angle θ. Results of this test for flight point 2, along with the previous results obtained using the Genesis initial guesses, are shown in Table 3.2. Trim values resulting from the arbitrary initial guesses are significantly different from those with the Genesis values, especially for the pitch angle θ, the flight path angle γ, and the throttle setting δ_{PLA}. Note also that trim has not been achieved at level flight, based on the nonzero values for γ.

Table 3.2: *Evaluation of* `Trim` *Reliability (FP2)*

Parameter	Units	Initial Guess (Genesis)	Trim Values	Initial Guess (Arbitrary)	Trim Values
V	ft/s	497.32	497.32	497.32	497.35
α	rad	0.1891	0.1853	0.1891	0.1884
q	rad/s	0.0	0.0	0.0	0.0
θ	rad	0.1891	0.1898	0.1891	0.0713
p	rad/s	0.0	0.0	0.0	0.0
ϕ	rad	0.0	0.0	0.0	0.0
r	rad/s	0.0	0.0	0.0	0.0
ψ	rad	0.0	0.2197	0.0	0.0237
β	rad	0.0	0.0	0.0	0.0
δ_H	deg	-7.059	-6.970	0.0	-7.123
δ_{PLA}	deg	78.95	78.83	30.00	30.05
δ_A	deg	0.0	0.0	0.0	0.0
δ_R	deg	0.0	0.0	0.0	0.0
γ	rad	0.0	0.0045	0.0	-0.1170

Although the initial guesses for the elevator of zero degrees and for the throttle of 30° are significantly different from the Genesis values, the elevator trim value of

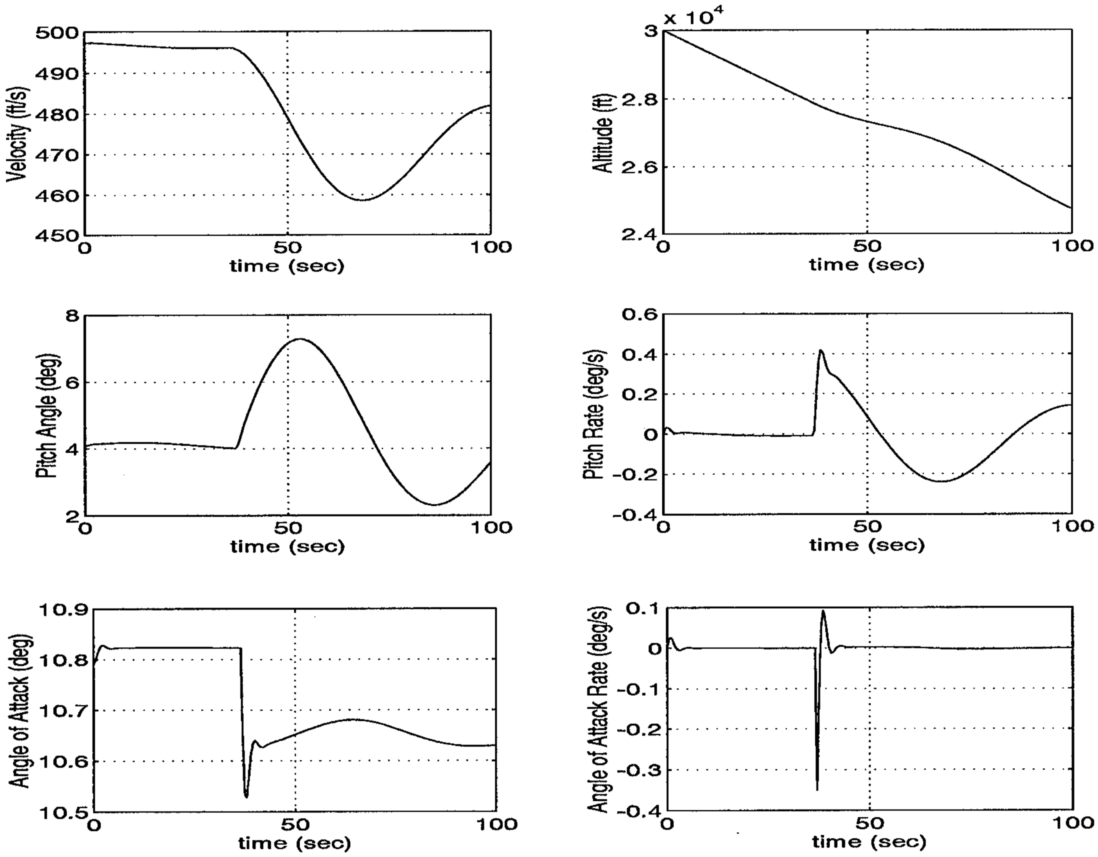


Figure 3.4: Aircraft Responses at FP2 using Trim Conditions Obtained from an Arbitrary Initial Guess (Table 3.2).

−7.123° converges roughly to the Genesis trim value of −6.970°. The trim throttle setting, however, remains virtually unchanged from the initial guess of 30°, resulting in an under-powered condition. The `trim` function attempts to correct for this by lowering the aircraft nose, allowing the aircraft to maintain velocity but causing the level flight constraint ($\gamma = 0$) to be violated. This situation is evident from Figure 3.4, showing the aircraft open-loop responses at this trim condition. In order for the `trim` routine to be useful under such conditions, it must provide a capability to *rigidly* set constraints on the aircraft output variables such as constant velocity and zero flight-path angle. Currently, the `trim` routine allows specification of *desired* values for output, state, and input variables, but these values are not held constant throughout the iteration. In both examples shown in Table 3.2, the flight path angle

γ was "set" to zero in `trim`, but in the final values both are nonzero at the equilibrium condition.

3.2 Evaluation of Linearized Model

From the MATLAB trim data shown previously in Table 3.1, a linearized state-space model at each flight point is determined using the MATLAB function `linmod`. To validate these models, open-loop responses of the nonlinear and linearized models are compared for 20-second pulse inputs applied independently to the elevator, the aileron, and the rudder controls. Although small variations exist, the linearized model responses for both flight points are reasonably close to those of the nonlinear model. State-space models of the full linearized equations of motion for both flight points are shown in Appendix C. Stability characteristics of the open-loop linearized models are summarized in Table 3.3.

3.2.1 Longitudinal Excitation

Elevator Pulse Input

The elevator pulse input for both flight conditions is 2° down for 20 seconds, resulting in an initial loss in altitude and an accompanying increase in airspeed. Figures 3.5-3.6 show the response plots for the velocity V, the altitude h, the pitch rate q, and the pitch angle θ at flight points 1 and 2 respectively.

3.2.2 Lateral-Directional Excitation

Aileron Pulse Input

The aileron input is a 20-second pulse of 1° for flight point 1 and a 20-second pulse of 2° for flight point 2. In both cases, the left aileron is up and the right aileron down, producing a left turn. The maximum roll angle reached is approximately 50° for both flight conditions. Figures 3.7-3.8 show the response plots for the angle of sideslip β, the yaw rate r, the roll rate p, and the roll angle ϕ for flight points 1 and 2 respectively.

Table 3.3: *Open-Loop Stability Characteristics*

Flight Point 1 Mode	Eigenvalues	Damping	Frequency (rad/s)
Altitude	0.0000	1.0000	0.0000
Spiral	-0.0015	1.0000	0.0015
Heading	-0.0325	1.0000	0.0325
Phugoid	$-0.0055 \pm 0.0805i$	0.0687	0.0807
Roll	-2.1211	1.0000	2.1211
Dutch Roll	$-0.4128 \pm 2.5913i$	0.1573	2.6239
Short Period	$-1.6407 \pm 2.2486i$	0.5894	2.7835
Flight Point 2 Mode	Eigenvalues	Damping	Frequency (rad/s)
Altitude	0.0000	1.0000	0.0000
Spiral	-0.0035	1.0000	0.0035
Heading	-0.0452	1.0000	0.0452
Phugoid	$-0.0072 \pm 0.0905i$	0.0798	0.0908
Roll	-0.7086	1.0000	0.7086
Dutch Roll	$-0.3461 \pm 1.9067i$	0.1786	1.9378
Short Period	$-0.7522 \pm 1.4587i$	0.4583	1.6412

Rudder Pulse Input

A pulse input to the twin rudders of 1° right is applied for 20 seconds at both flight conditions. The result is a right turn, with a maximum roll angle for flight conditions 1 and 2 of approximately 70° and 80° respectively. Figures 3.9-3.10 show the response plots for the sideslip angle β, the yaw rate r, the roll rate p, and the roll angle ϕ at flight points 1 and 2 respectively.

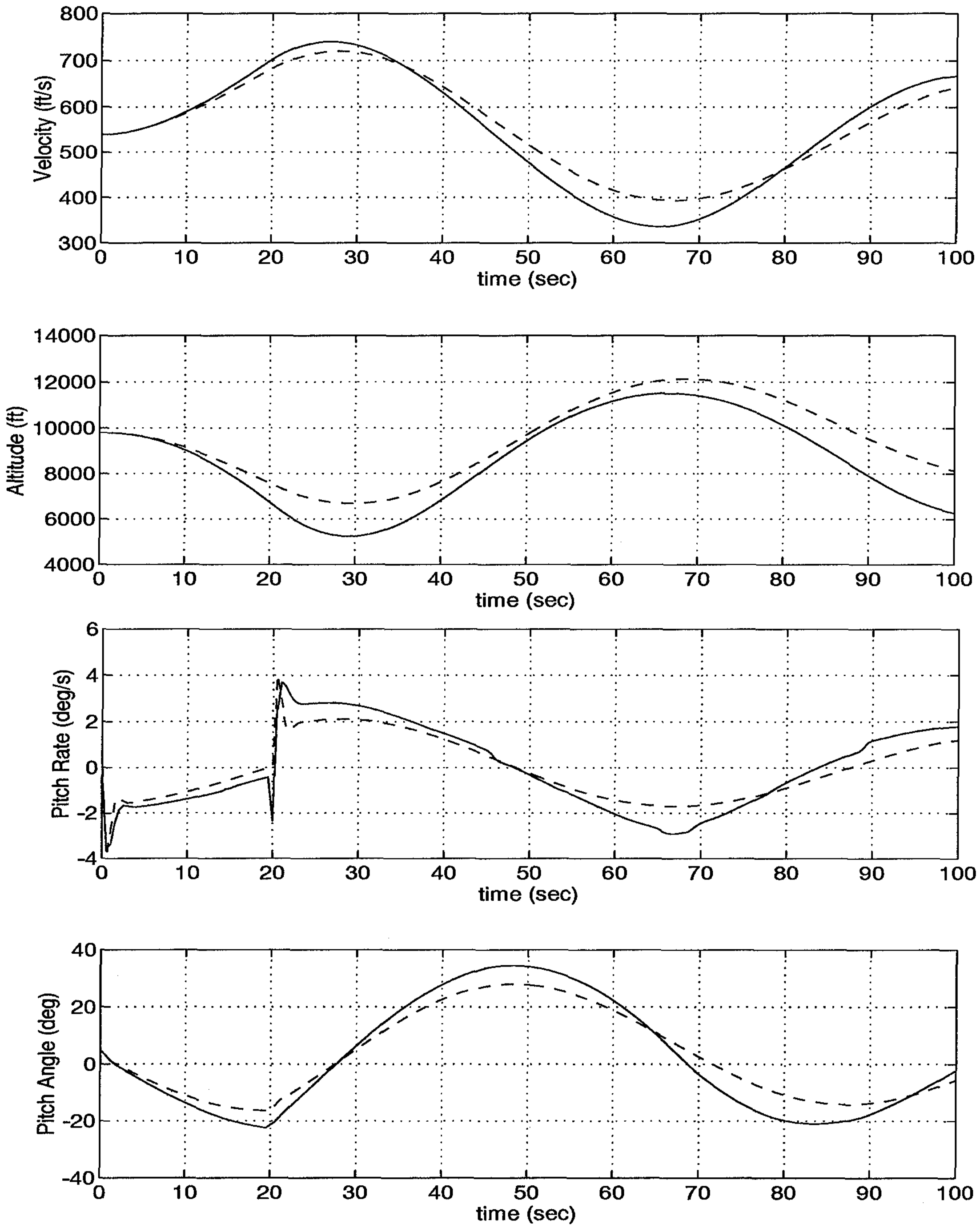


Figure 3.5: Flight Point 1—Aircraft Responses to a 20-second Elevator Pulse of 2°: Nonlinear Model (solid line) and Linearized Model (dashed line).

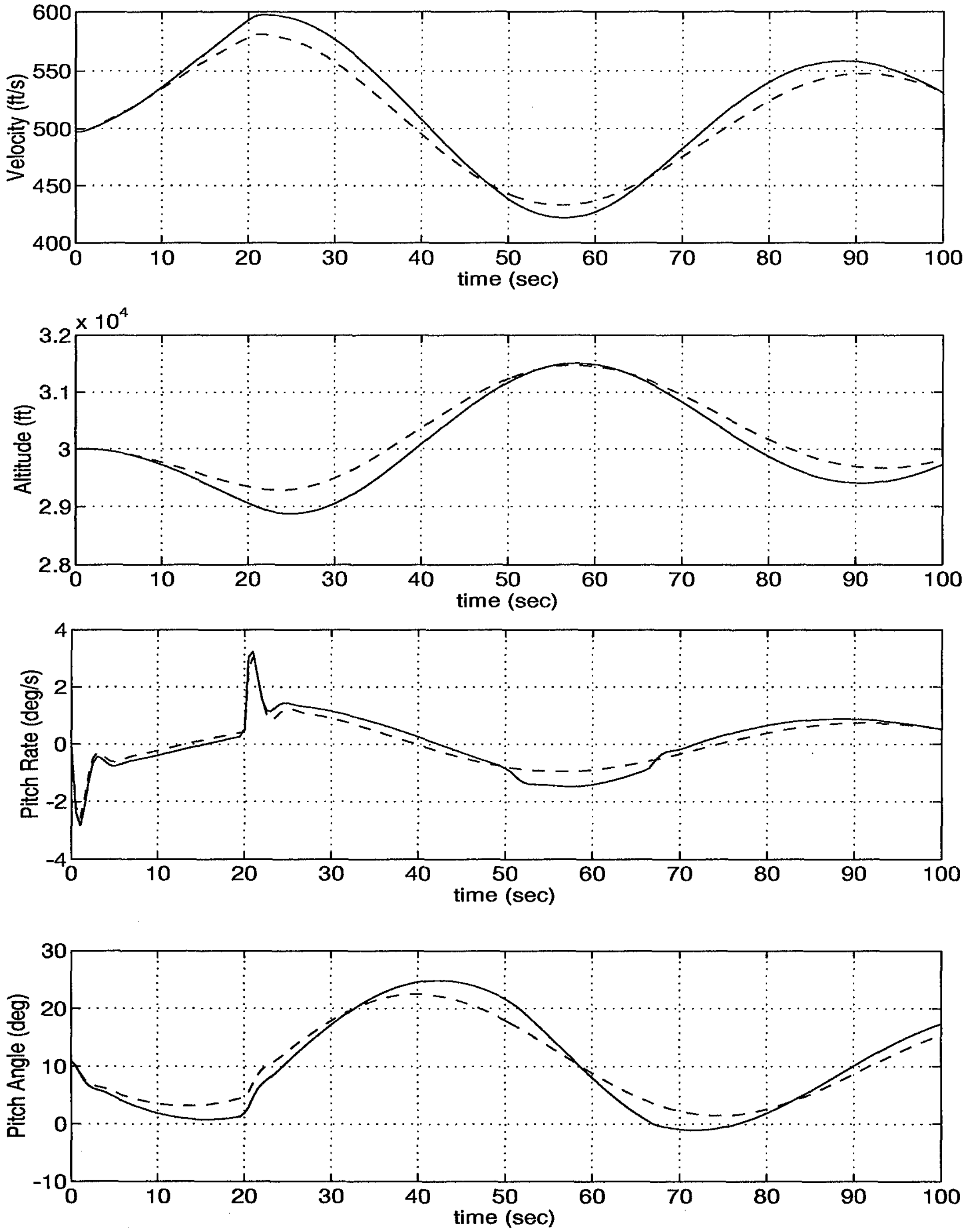


Figure 3.6: Flight Point 2—Aircraft Responses to a 20-second Elevator Pulse of 2°: Nonlinear Model (solid line) and Linearized Model (dashed line).

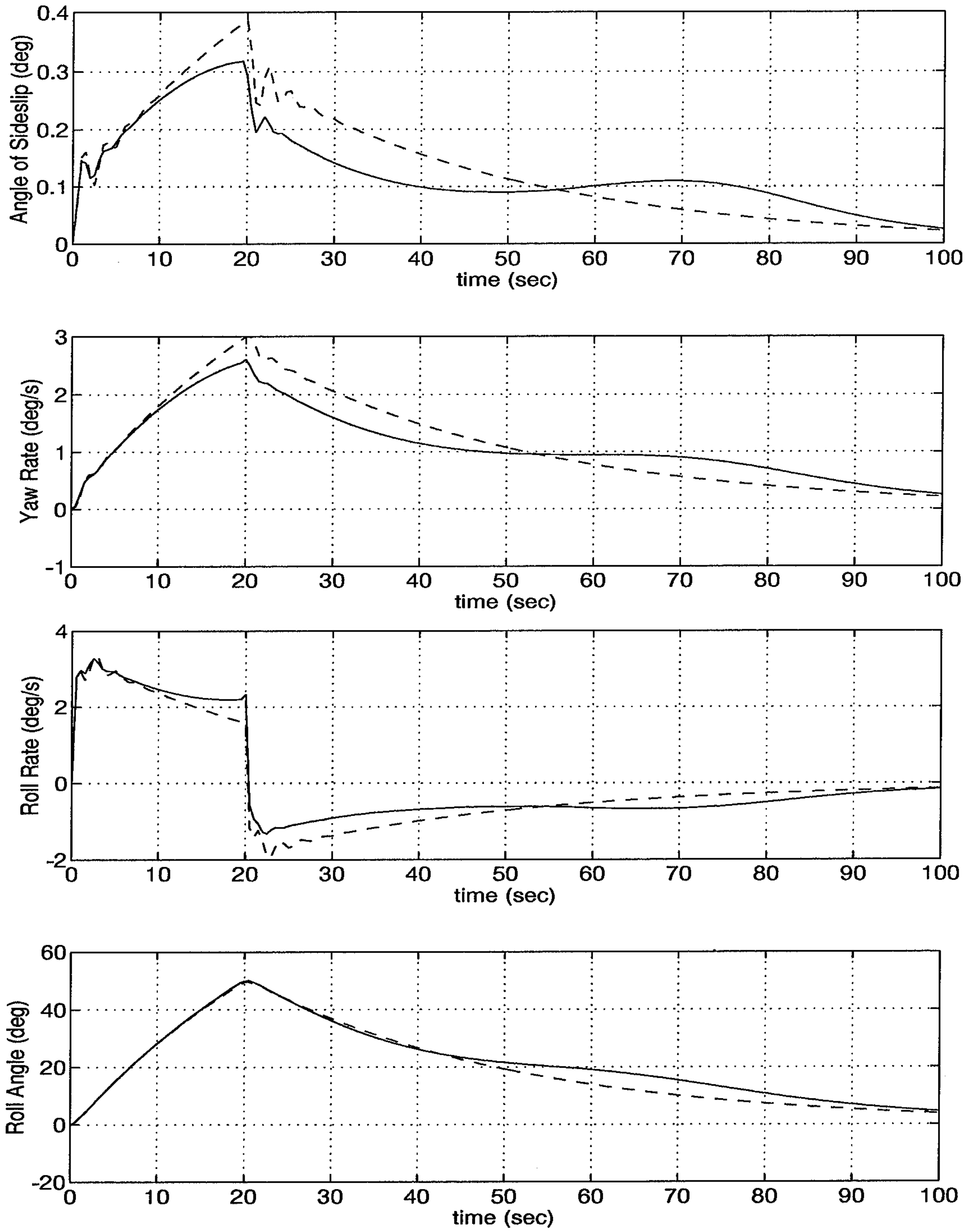


Figure 3.7: Flight Point 1—Aircraft Responses to a 20-second Aileron Pulse of 1°: Nonlinear Model (solid line) and Linearized Model (dashed line).

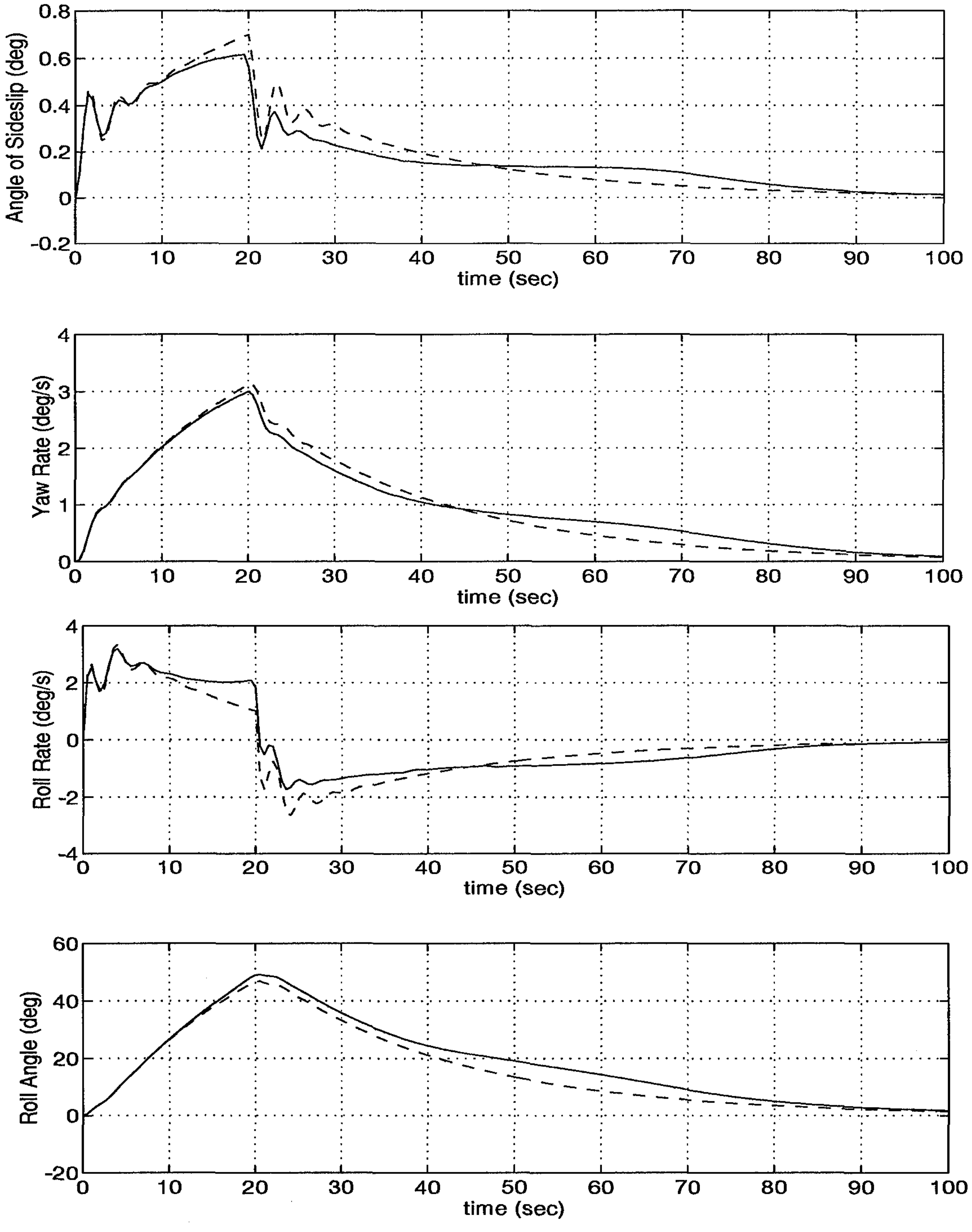


Figure 3.8: Flight Point 2—Aircraft Responses to a 20-second Aileron Pulse of 2°: Nonlinear Model (solid line) and Linearized Model (dashed line).

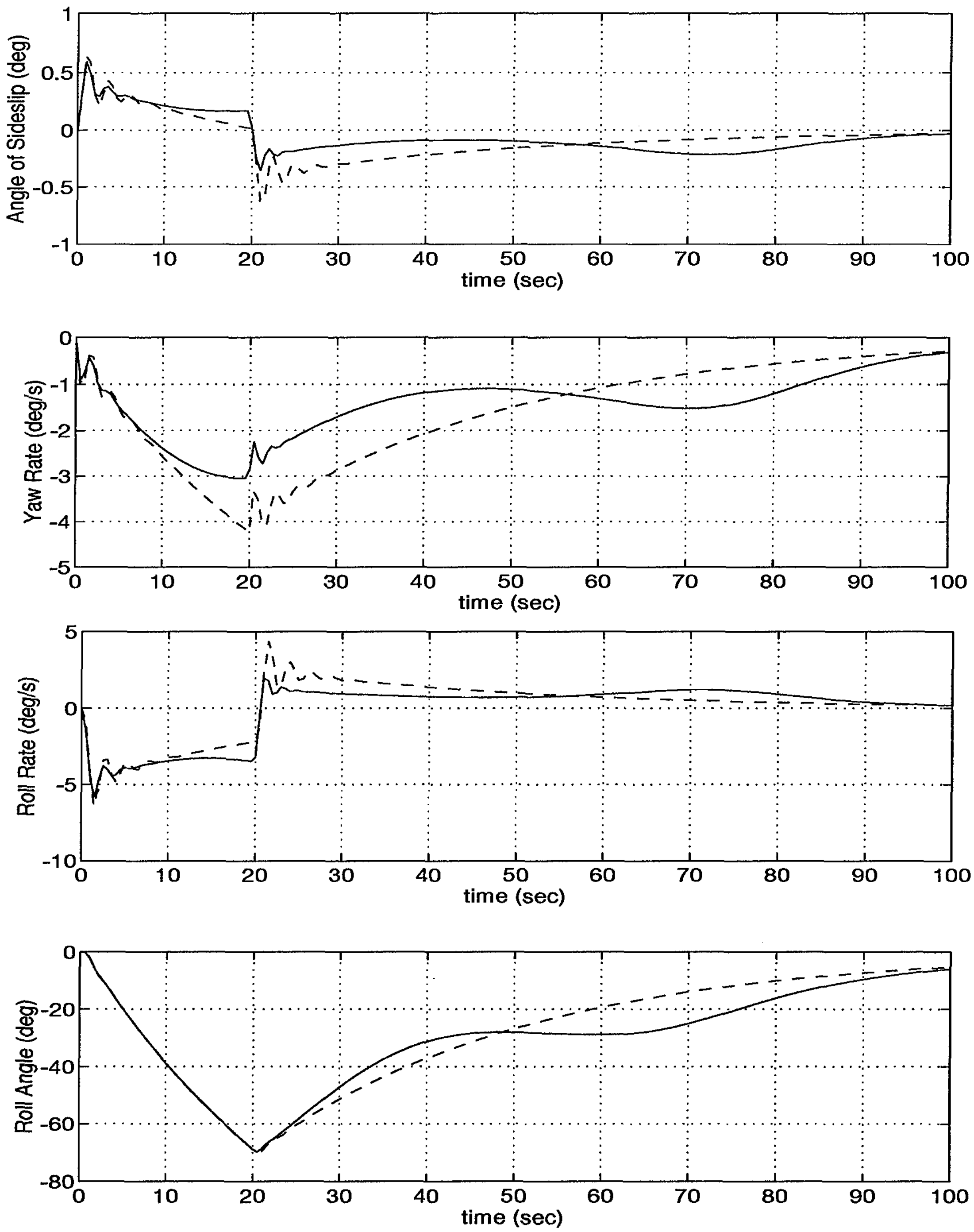


Figure 3.9: Flight Point 1—Aircraft Responses to a 20-second Rudder Pulse of 1°: Nonlinear Model (solid line) and Linearized Model (dashed line).

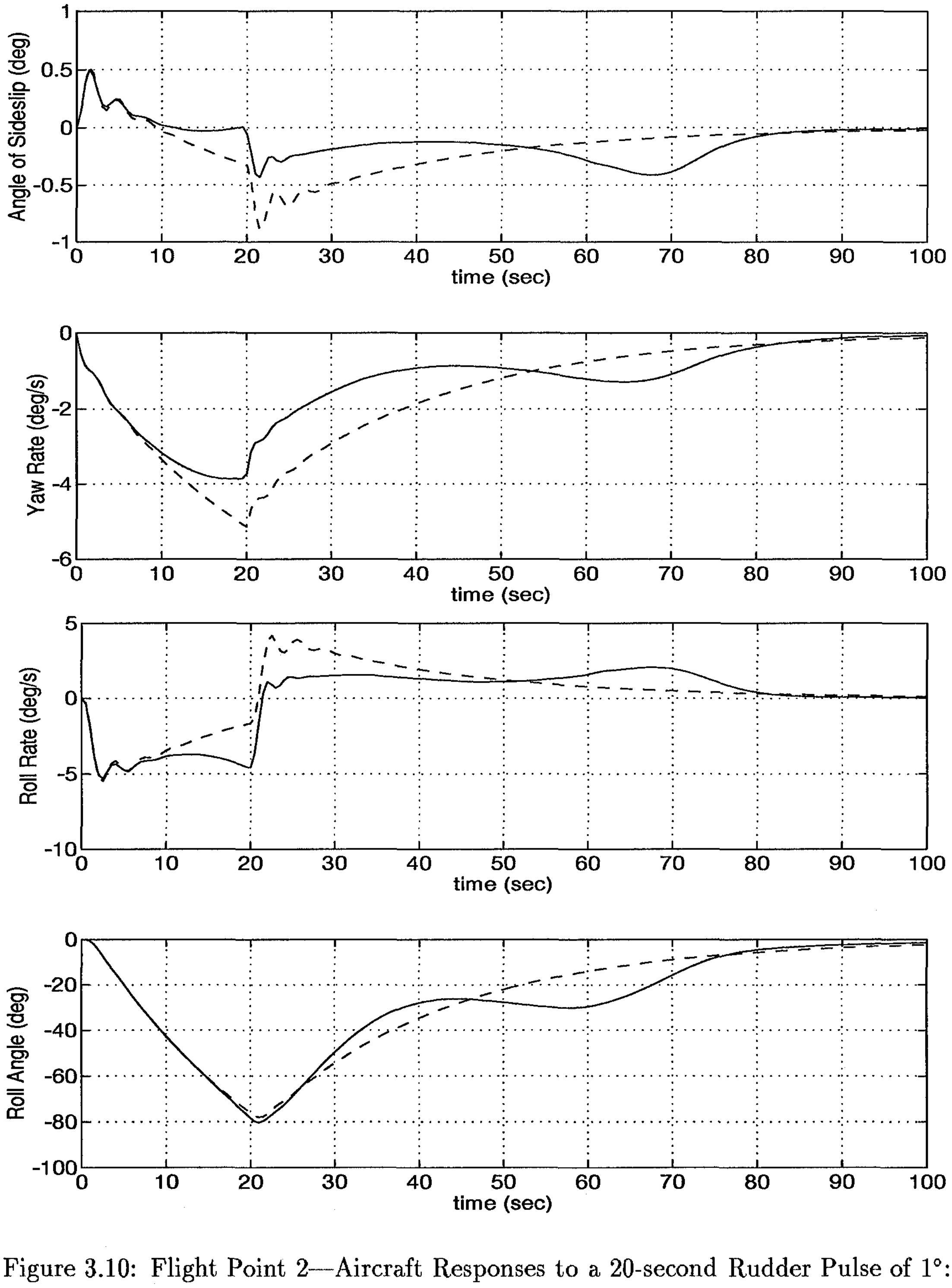


Figure 3.10: Flight Point 2—Aircraft Responses to a 20-second Rudder Pulse of 1°: Nonlinear Model (solid line) and Linearized Model (dashed line).

Chapter 4

LONGITUDINAL CONTROL USING TECS

4.1 Background

Since the early years of aircraft automatic flight control system design, the fundamental approach to autopilot design has evolved around the single-loop feedback structure. Although this method has proven satisfactory for the time, recently it has been found that the inherent limitations of this design method are holding back the development of more capable flight control systems. Extensive studies such as discussed in [3] demonstrate that the conventional design methods have reached their fundamental limits. Future improvements therefore rely on the development of new system architectures.

The Total Energy Control System (TECS) was developed in response to this observation. The conventional flight control system is fundamentally limited by its single-loop development of the throttle- and elevator-command loops, which neglects the cross-coupling effects of the longitudinal dynamics. The TECS design, however, has the fundamental objective of integrating flight path and speed control based on "energy compensation" techniques. In the conventional design, flight-path control is achieved exclusively through feedback to the elevator while speed control is achieved using feedback to the throttles. One of the most serious design deficiencies of this method is the potential for adverse cross-coupling between the elevator and throttle controls once both feedback loops are closed. Both stability and performance of the entire system may be adversely affected. Similar design deficiencies exist for conventional lateral control systems.

By focusing on integration between the flight path and speed control systems, the TECS design has incorporated the following features to overcome this limitation: the use of a multiloop control structure with crossfeed paths between the speed and flight path to both the throttle and the elevator, the use of total energy rate and total energy distribution rate in the control design, and the use of command paths for control of flight path and longitudinal acceleration. The reader is referred to the following sources for further background in the methods of total energy control [2, 4, 7, 8].

4.2 Development of the TECS Concept

Numerous methods exist for simultaneous operation of the throttle and elevator using total energy control methods. One such method is based on the understanding that the throttle is fundamentally related to energy rate, in that it either increases or decreases the overall system energy. The elevator, on the other hand, is related to energy distribution. It causes an exchange between kinetic and potential energy in the system by causing the aircraft to either climb or descend. An example of the advantage offered by the TECS method is with the simultaneous commands of increased flight-path angle and decreased speed. In a single-loop design, both the elevator and throttle would respond, possibly resulting in over-control or the two controls "opposing" one another. Using the total energy control concept, the system recognizes that a simultaneous climb and decrease in airspeed is essentially an exchange of the aircraft kinetic energy for the potential energy, and would therefore rely primarily on the elevator to achieve the desired response.

In mathematical terms, the total energy control concept is fundamentally based on the total energy $E(t)$ of the aircraft. The following summarizes the derivation of the total energy control concept in [9]. For a point mass, $E(t)$ is described in terms of the aircraft mass m, the total velocity $V(t)$, the altitude $h(t)$, and the gravitational acceleration g as

$$E(t) = \frac{1}{2}mV(t)^2 + mgh(t) \tag{4.1}$$

By differentiating equation (4.1) with respect to time, the total energy rate $\dot{E}(t)$ is

$$\dot{E}(t) = mgV(t)\left(\frac{\dot{V}(t)}{g} + \gamma(t)\right) \tag{4.2}$$

where the flight-path angle $\gamma(t)$ is in radians. The thrust required $T_{req}(t)$ is then derived from the equations of motion along the flight path. By assuming the initial thrust offsets the drag, and that variations in drag are slow, then the thrust required for a particular level of total energy rate is

$$T_{req}(t) = \frac{\dot{E}(t)}{V(t)} = mg\left(\frac{\dot{V}(t)}{g} + \gamma(t)\right) \tag{4.3}$$

From equation (4.3), it is evident that the aircraft total energy state is determined by the thrust required, which is controlled by the throttle. However, the capacity to distribute energy between its kinetic and potential states is not well-handled by the throttle. Because the elevator causes relatively little drag while controlling the aircraft angle of attack for small inputs, it is an energy-conserving mechanism. Rather than adding to the total system energy as with the throttle, the elevator essentially distributes the energy between the kinetic and the potential states while maintaining the total energy constant. Thus, by using proportional and integral control on the total energy rate and the energy distribution rate, the throttle and the elevator commands are

$$\begin{aligned} \delta_{T_c}(s) &= mg\left(K_{TP} + \frac{K_{TI}}{s}\right)\left(\frac{\dot{V}_\epsilon(s)}{g} + \gamma_\epsilon(s)\right) && (4.4) \\ \delta_{e_c}(s) &= K_\gamma\left(K_{EP} + \frac{K_{EI}}{s}\right)\left(\frac{\dot{V}_\epsilon(s)}{g} + \gamma_\epsilon(s)\right) && (4.5) \end{aligned}$$

where the flight-path error $\gamma_\epsilon(s)$ and the acceleration error $\dot{V}_\epsilon(s)$ are

$$\begin{aligned} \gamma_\epsilon(s) &= \gamma(s) - \gamma_c(s) \\ \dot{V}_\epsilon(s) &= \dot{V}(s) - \dot{V}_c(s) \end{aligned}$$

The proportional feedback gains to the throttle and the elevator are K_{TP} and K_{EP}, and the integral feedback gains are K_{TI} and K_{EI} respectively.

The overall closed-loop system using TECS is shown in Figure 4.1. Because the model is designed to be used for the generic aircraft, the inputs have been reduced to elevator, throttle, flaps, rudder, and aileron. In the case of the F-15, the elevator is represented by the symmetric stabilator and the throttle by the symmetric PLA. The differential stabilator and the individual PLA settings are left out to maintain generality. Flaps are not defined in the F-15 dynamic model. Inputs to the longitudinal TECS controller are normalized acceleration $\dot{V}_\epsilon(s)/g$, velocity V, altitude h, flight-path angle γ, pitch rate q and pitch angle θ. Inputs to the lateral TECS controller are roll rate p, roll angle ϕ, sideslip rate $\dot{\beta}$, sideslip angle β, yaw angle ψ, yaw rate r and heading rate of change $\dot{\psi}$. The specific TECS design for the longitudinal dynamics will be discussed next. For more information on the lateral TECS design, the reader is referred to [9].

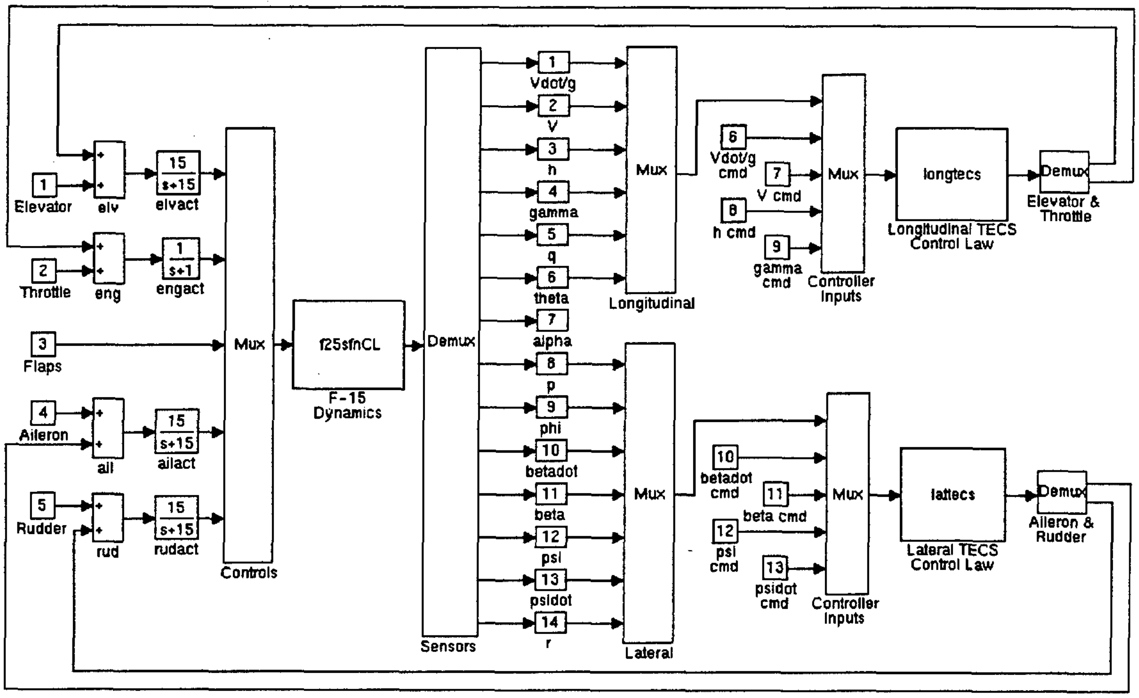


Figure 4.1: Closed-Loop Model Block Diagram for TECS Control Law.

4.3 Longitudinal TECS Structure

The longitudinal TECS controller is composed of commands to altitude h_c, flight-path angle γ_c, normalized acceleration $\dot{V}_c/g$, and velocity V_c. The controller can operate with h_c and V_c specified only, from which the γ_c and $\dot{V}_c/g$ values are derived using the proportional gains K_h and K_v respectively. Otherwise, the values for γ_c and $\dot{V}_c/g$ can be specified directly. The structure of the longitudinal TECS controller is shown in Figure 4.2. Note that the integral action on the airspeed and altitude are accomplished through feedback of the velocity and altitude errors to the acceleration

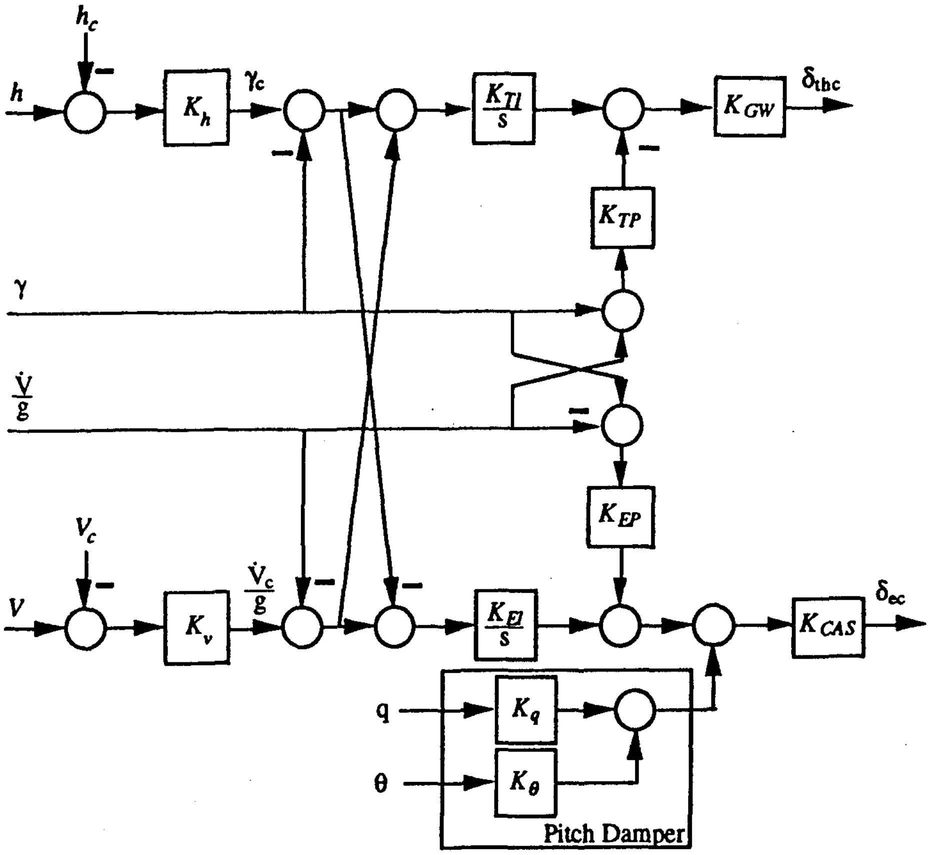


Figure 4.2: Longitudinal TECS Controller Structure.

and flight-path commands, respectively.

A state-space representation for the longitudinal TECS controller, shown in the format of [9], is

$$\begin{cases} \dot{x}_{c_{long}} &= A_{c_{long}} x_{c_{long}} + B_{c_{long}} y_{s_{long}} \\ \delta_{c_{long}} &= C_{c_{long}} x_{c_{long}} + D_{c_{long}} y_{s_{long}} \end{cases}$$

where the controller state vector $x_{c_{long}}$, the measurement vector $y_{s_{long}}$ and the control input vector $\delta_{c_{long}}$ are

$$\begin{aligned} x_{c_{long}} &= [x_{c_1}(t), x_{c_2}(t)]^T \\ y_{s_{long}} &= [\Delta\gamma(t), \Delta\dot{V}(t)/g, \Delta q(t), \Delta\theta(t), \Delta V(t), \Delta h(t)]^T \\ \delta_{c_{long}} &= [\Delta\delta_{thc}(t), \Delta\delta_{ec}(t)]^T \end{aligned}$$

and

$$
\begin{aligned}
A_{c_{long}} &= \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix} \\
B_{c_{long}} &= \begin{bmatrix} -1 & -1 & 0 & 0 & K_v & K_h \\ 1 & -1 & 0 & 0 & K_v & -K_h \end{bmatrix} \\
C_{c_{long}} &= \begin{bmatrix} K_{GW}K_{TI} & 0 \\ 0 & K_{CAS}K_{EI} \end{bmatrix} \\
D_{c_{long}} &= \begin{bmatrix} K_{GW}K_{TP} & K_{GW}K_{TP} & 0 & 0 & 0 & 0 \\ K_{CAS}K_{EP} & -K_{CAS}K_{EP} & K_{CAS}K_q & K_{CAS}K_\theta & 0 & 0 \end{bmatrix}
\end{aligned}
$$

The parameters K_{TP}, K_{TI}, K_{EP} and K_{EI} are the proportional and integral gains on the throttle and the elevator controls. The gains K_q and K_θ essentially form the stability augmentation system (SAS), while the gains K_v and K_h provide feedback correction on the airspeed and altitude errors, respectively. The gains K_{CAS} and K_{GW} are scheduled according to the aircraft calibrated airspeed V_{CAS} and gross weight W, where $W = mg$. Determination of the above gains will be accomplished using the constrained parameter optimization program SANDY. The SIMULINK model for the longitudinal TECS controller is shown in Figure 4.3. Implementation of the controller with the F-15 nonlinear model is the topic of the next section.

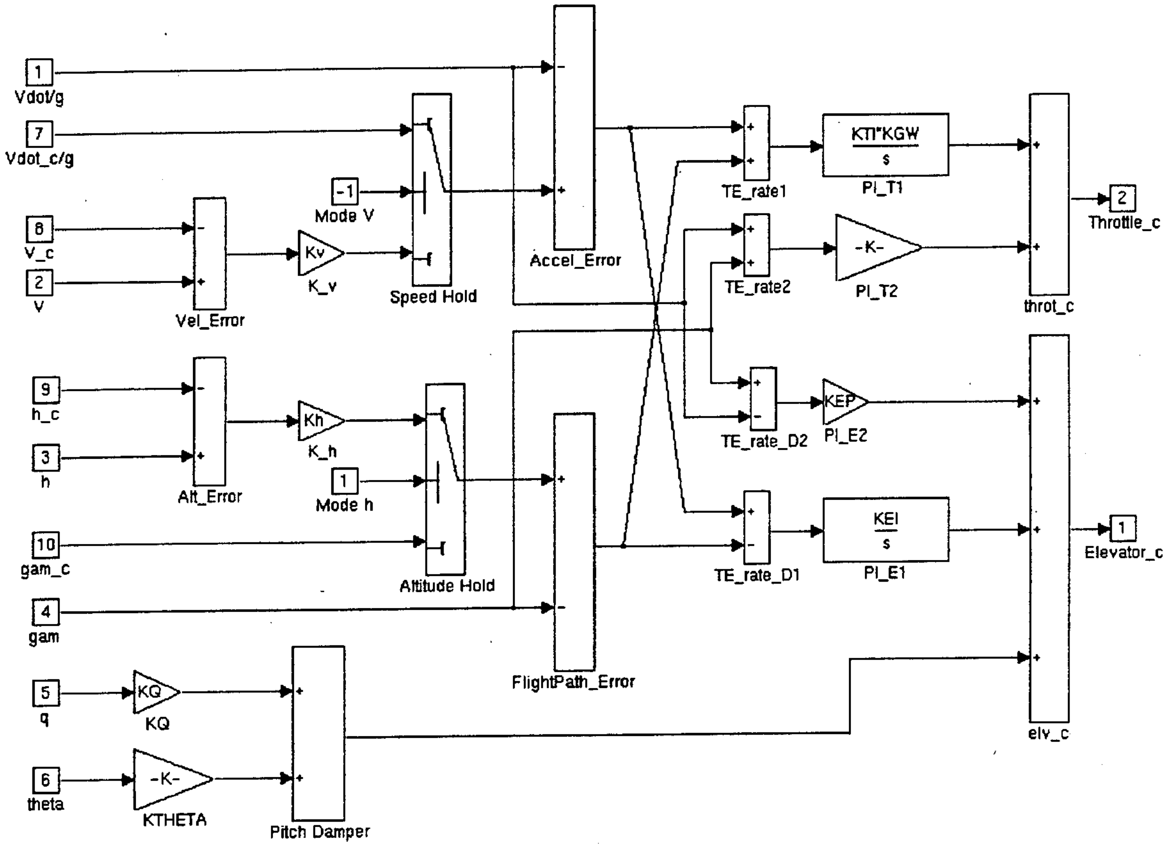


Figure 4.3: SIMULINK Model for the Longitudinal TECS Controller.

Chapter 5

TECS CONTROLLER PERFORMANCE

The primary objective of this section is to evaluate a control law designed for the F-15 using the linearized aircraft model and then to validate this control law by implementing it on the nonlinear model. In this example, the control law used is the longitudinal TECS controller and the gains have been selected using the numerical optimization program SANDY. Describing the controller design is not the intent of this section. The gains selected for each flight point are shown in Appendix D.

5.1 Linearized Closed-Loop Model Evaluation

Validity of the linearized models at the two flight conditions was established in Chapter 3. Now we will evaluate the closed-loop system using the linearized aircraft dynamics.

5.1.1 Closed-Loop Characteristics

Stability of the system is evaluated initially from the closed-loop eigenvalues. Table 5.1 shows the closed-loop eigenvalues of lowest damping and frequency for each of the flight conditions. For both flight points, the minimum damping is 0.7 and the closest pole to the origin is at -0.100. A stable system should therefore result. The complete set of closed-loop eigenvalues is shown in Appendix D.

Robustness of the system is evaluated for single-loop stability margins. The elevator and throttle control loops are "broken" independently and the resulting gain and phase margins are determined. Figures 5.1-5.2 show the Bode plots with the elevator-loop and throttle-loop broken respectively for the two flight conditions. As seen in Table 5.2, the lowest gain and phase margins substantially exceed even the conservative minimum requirements of ± 6 dB and $\pm 45°$ respectively, indicating that adequate robustness exists.

Table 5.1: *Closed-Loop Stability Characteristics*

Flight Point	Eigenvalues	Damping	Frequency (rad/s)
1	$-0.357 \pm 0.364i$	0.700	0.510
	-0.100	1.000	0.100
2	$-0.124 \pm 0.126i$	0.700	0.176
	-0.100	1.000	0.100

Table 5.2: *Single-Loop Stability Margins*

Flight Point	δ_{DH} G.M. (dB)	δ_{DH} P.M. (deg)	δ_{PLA} G.M. (dB)	δ_{PLA} P.M. (deg)
1	25.82	-64.3	∞	-60.9
2	-44.19	-62.3	∞	-62.3

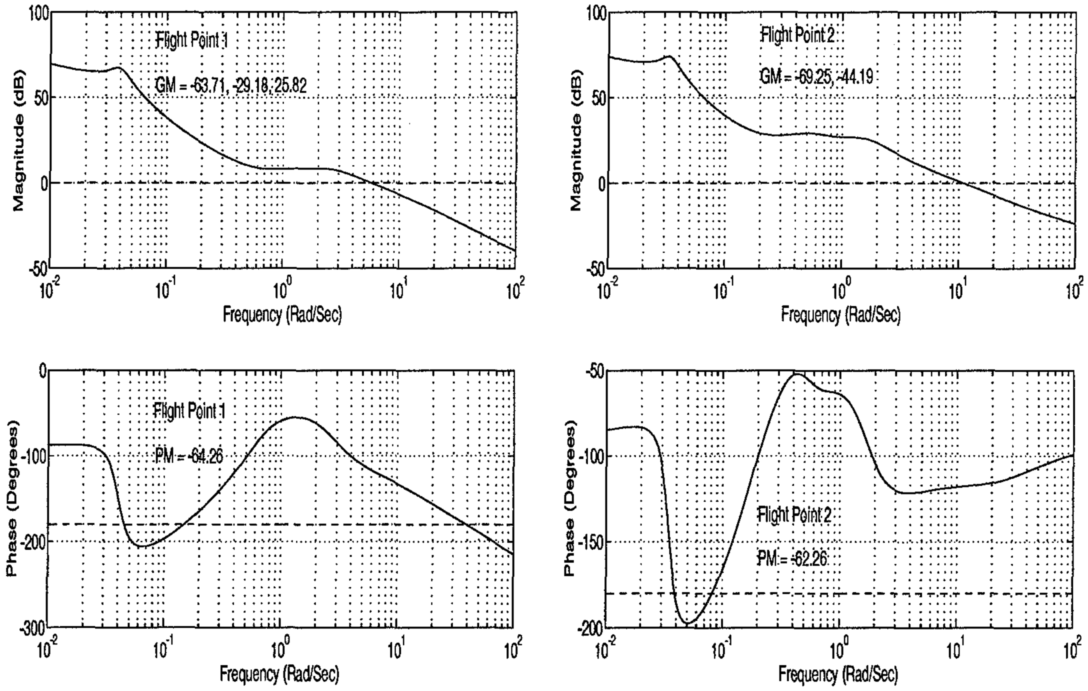


Figure 5.1: Elevator Control-Loop Bode Plots.

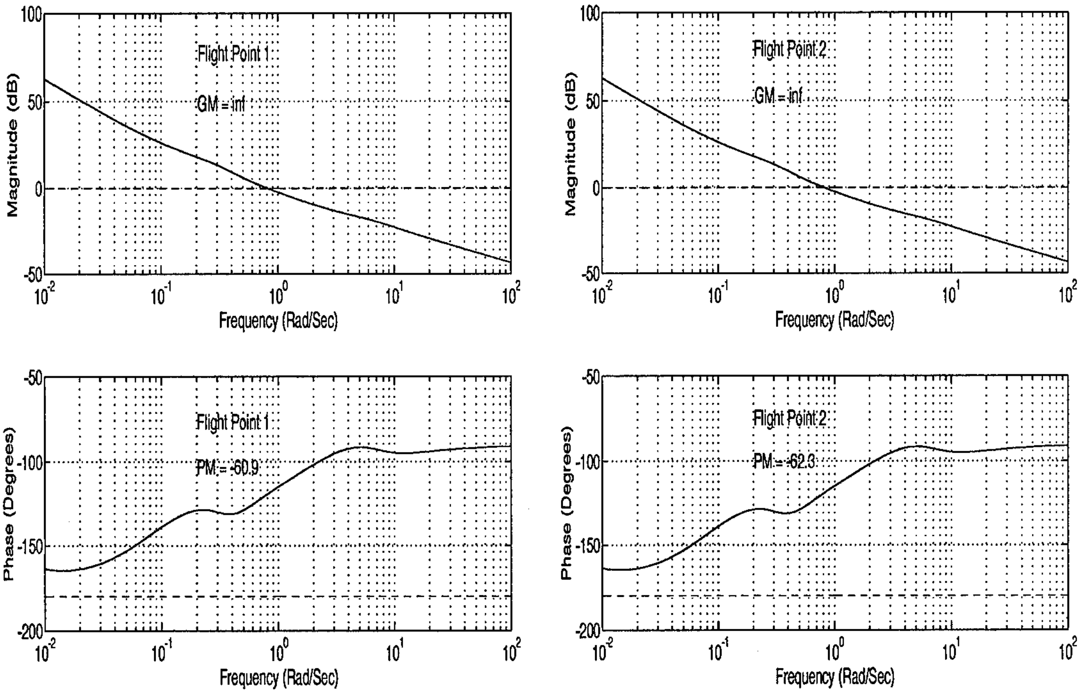


Figure 5.2: Throttle Control-Loop Bode Plots.

The ability of the closed-loop system to reject disturbances is also evaluated. As shown in Table 5.3, the root-mean-square (rms) values of the load factor at the aircraft center-of-gravity $n_{z_{CG}}$, the altitude h, the total velocity V, the symmetric stabilator δ_H and the throttle δ_{PLA} to turbulence of unit intensity in both the u- and w-directions are all satisfactorily small at both design flight points. The specification for $n_{z_{CG}}$ in a commercial aircraft is typically a maximum rms of 0.1 g. Although this requirement is satisfied here, it would likely be further relaxed for a fighter aircraft such as the F-15.

Table 5.3: *RMS Responses to Turbulence*

Flight	$n_{z_{CG}}$	h	V	δ_{DH}	δ_{PLA}
Point	(g)	(ft)	(ft/sec)	(deg)	(deg)
1	0.0143	4.93	0.0717	0.0432	0.2300
2	0.0092	6.17	0.0878	0.0793	0.7964

Command bandwidths for the velocity and altitude variables are also evaluated to determine if adequate control power exists for both the elevator and throttle. Figure 5.3 shows the Bode plots for the V/V_{cmd} and h/h_{cmd} transfer functions at flight points 1 and 2. The V/V_{cmd} bandwidth is 0.060 rad/s for flight point 1 and 0.058 rad/s for flight point 2. The h/h_{cmd} bandwidth is 0.060 rad/s for both flight conditions. Comparing the bandwidths of V/V_{cmd} to h/h_{cmd}, the values for flight point 1 are nearly identical while those for flight point 2 differ by 0.002 rad/s. In both cases, the differences in bandwidth are small enough that control power should be equally distributed to both the elevator and throttle.

Based on the above analysis, the closed-loop linearized system command responses should exhibit satisfactory stability, robustness, and disturbance rejection. These responses will be evaluated in the next section.

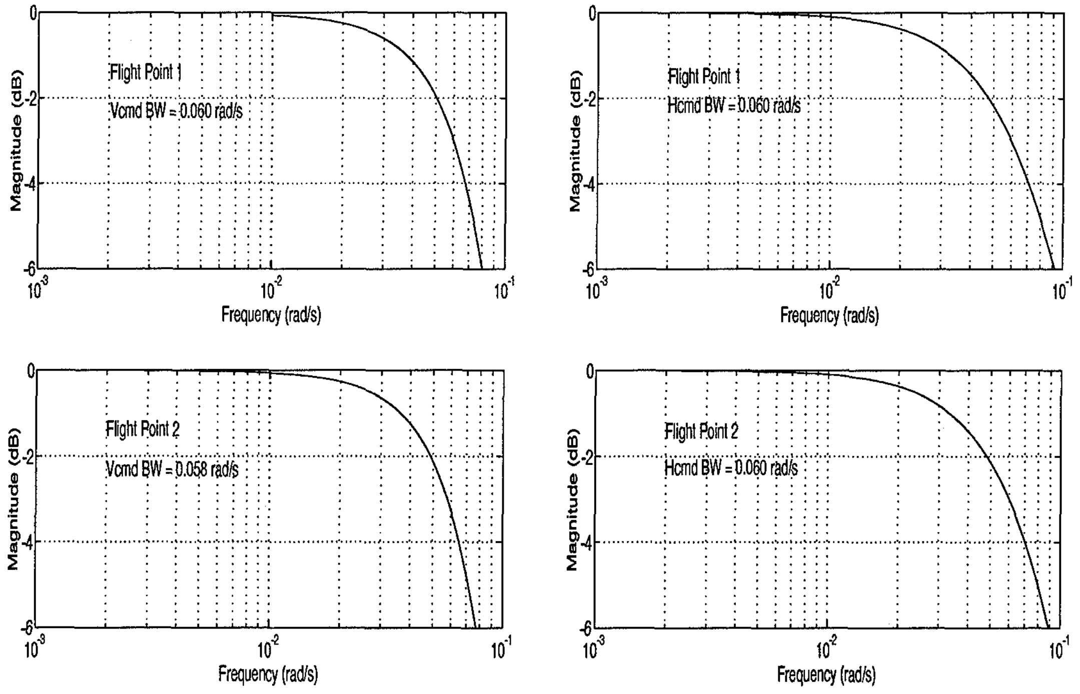


Figure 5.3: V_c and h_c Command Frequency Responses.

5.1.2 Command Responses

The closed-loop command responses to aircraft velocity and altitude are determined using the SIMULINK system shown in Chapter 4, with the linearized F-15 dynamics. Responses for the velocity V, altitude h, elevator δ_H and throttle δ_{PLA} are discussed below. The complete set of responses are shown in Appendix D.

Velocity Command

The linearized aircraft responses to a 20 ft/s velocity command are shown for both flight conditions in Figures 5.4-5.5. The velocity tracking is satisfactory, with a settling time (T_s) of under 50 seconds in both cases. At flight point 1, the excursion from the trim altitude is approximately 6 ft, whereas flight point 2 is approximately 47 ft. Both the elevator and throttle control responses for flight point 2 are more than twice those of flight point 1. Note also the slight oscillation in elevator which occurs in both conditions at approximately 10 seconds.

Altitude Command

The linearized aircraft responses to a 1000 ft altitude command are shown for both flight conditions in Figures 5.6-5.7. The altitude tracking shows almost a 10-second delay, resulting in a T_s of greater than 60 seconds for both flight points. However, the velocity deviation from trim is kept very small. This indicates possibly too little authority in elevator control. Again, flight point 1 has control responses of less than half those of flight point 2.

Combined Velocity and Altitude Commands

The linearized aircraft responses to simultaneous commands of 20 ft/s in velocity and 1000 ft in altitude are shown for both flight conditions in Figures 5.8-5.9. The most notable difference here from the two independent commands is the increase in the velocity T_s to greater than 50 seconds. For flight point 2, the velocity response nearly overshoots the command tracking at approximately 20 seconds. The same relationship in control response magnitudes for the two flight conditions exists here.

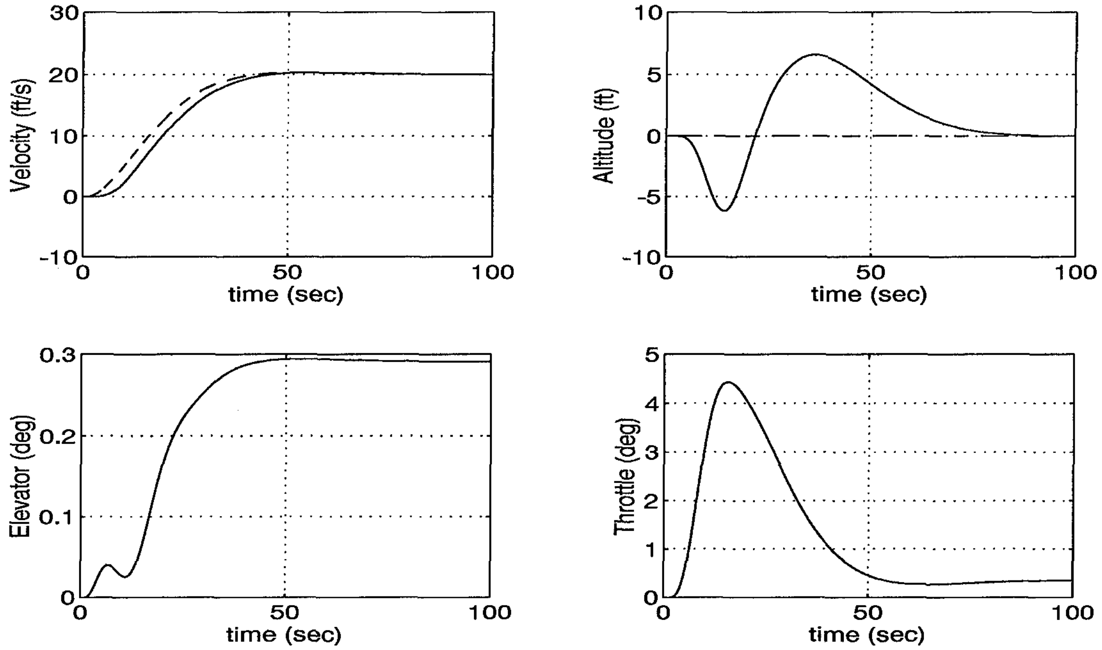


Figure 5.4: Flight Point 1—Linear Aircraft Responses to a 20 ft/s Velocity Command.

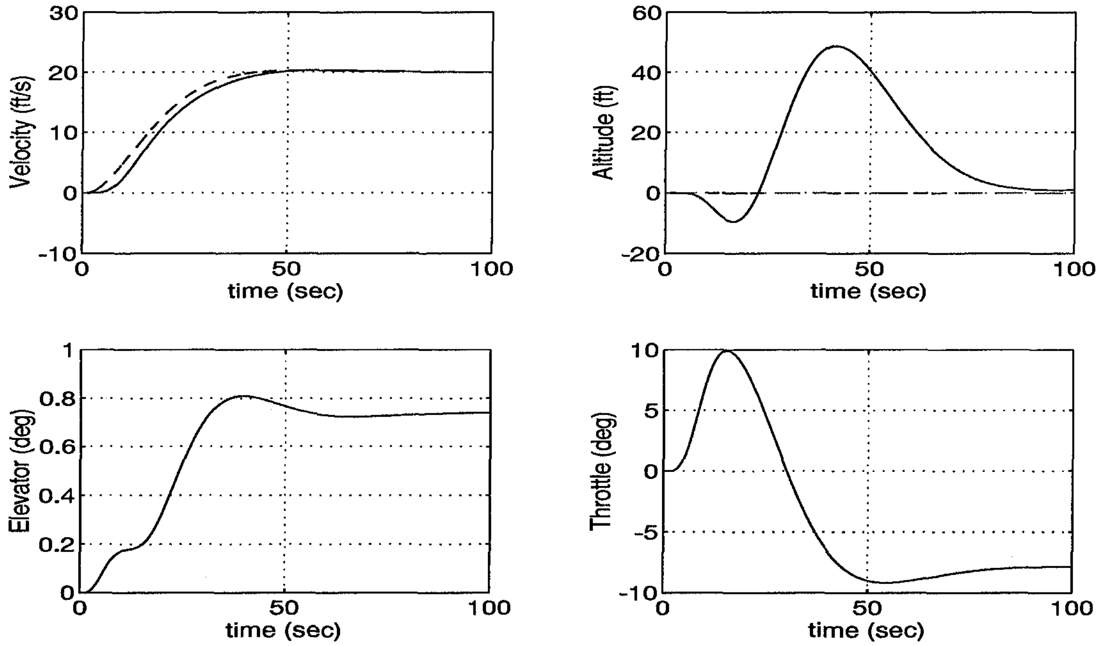


Figure 5.5: Flight Point 2—Linear Aircraft Responses to a 20 ft/s Velocity Command.

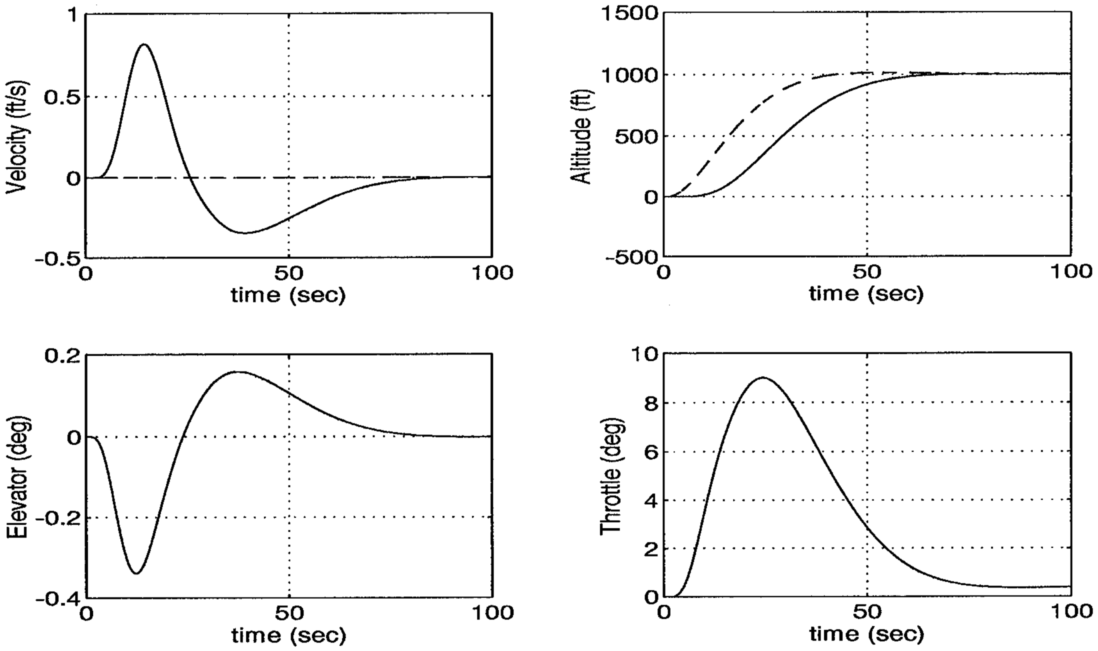


Figure 5.6: Flight Point 1—Linear Aircraft Responses to a 1000 ft Altitude Command.

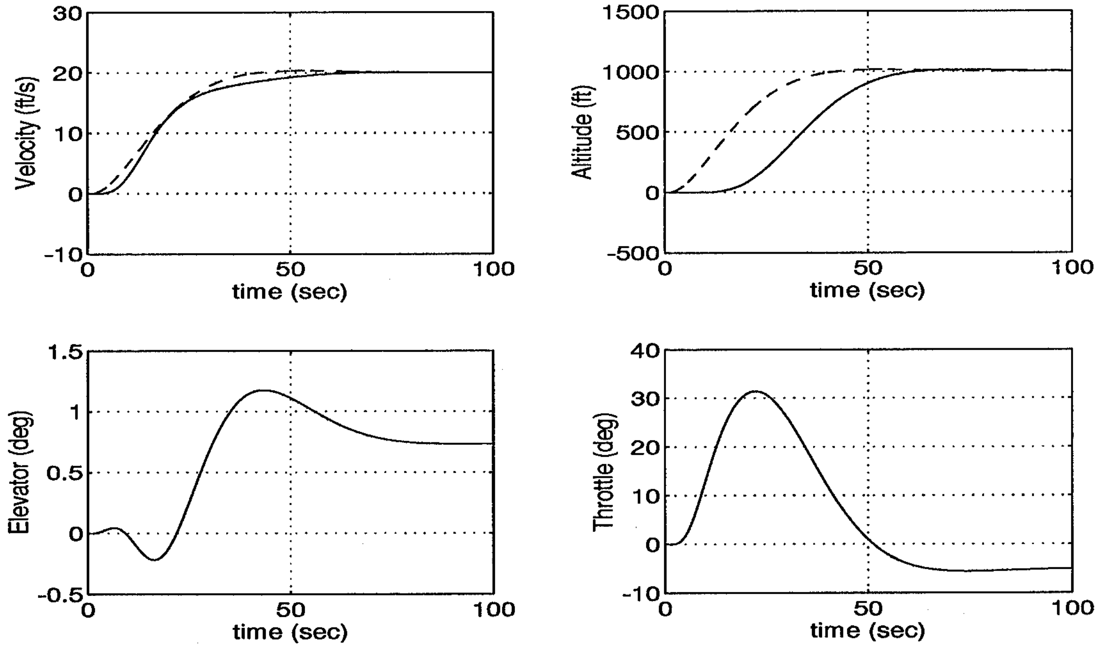


Figure 5.7: Flight Point 2—Linear Aircraft Responses to a 1000 ft Altitude Command.

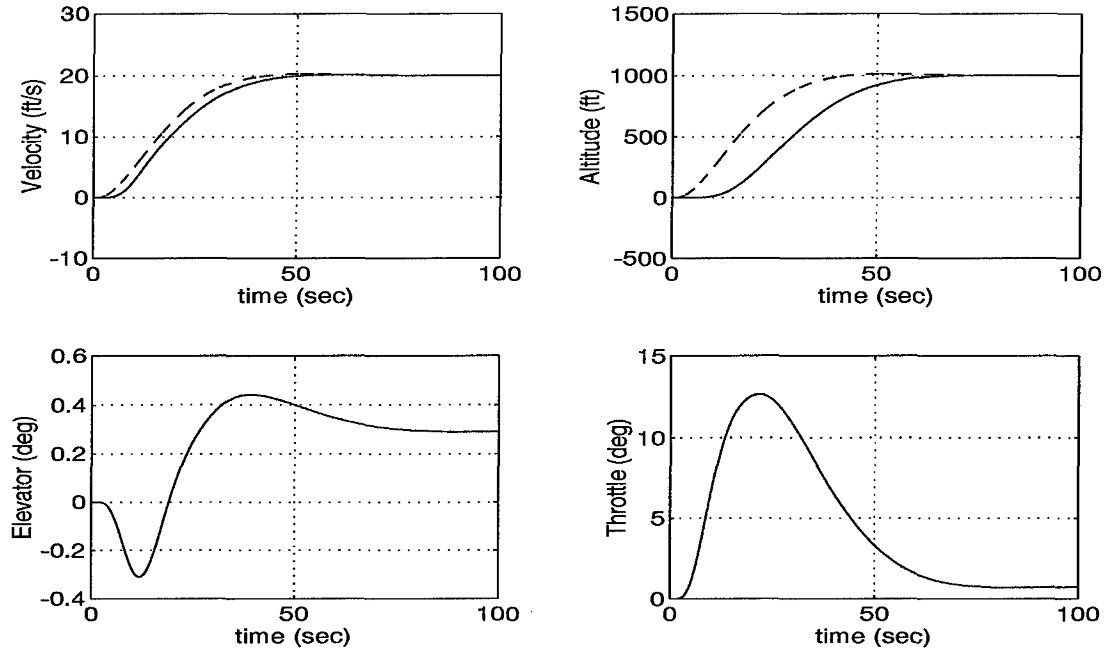


Figure 5.8: Flight Point 1—Linear Aircraft Responses to a 20 ft/s Velocity Command and 1000 ft Altitude Command.

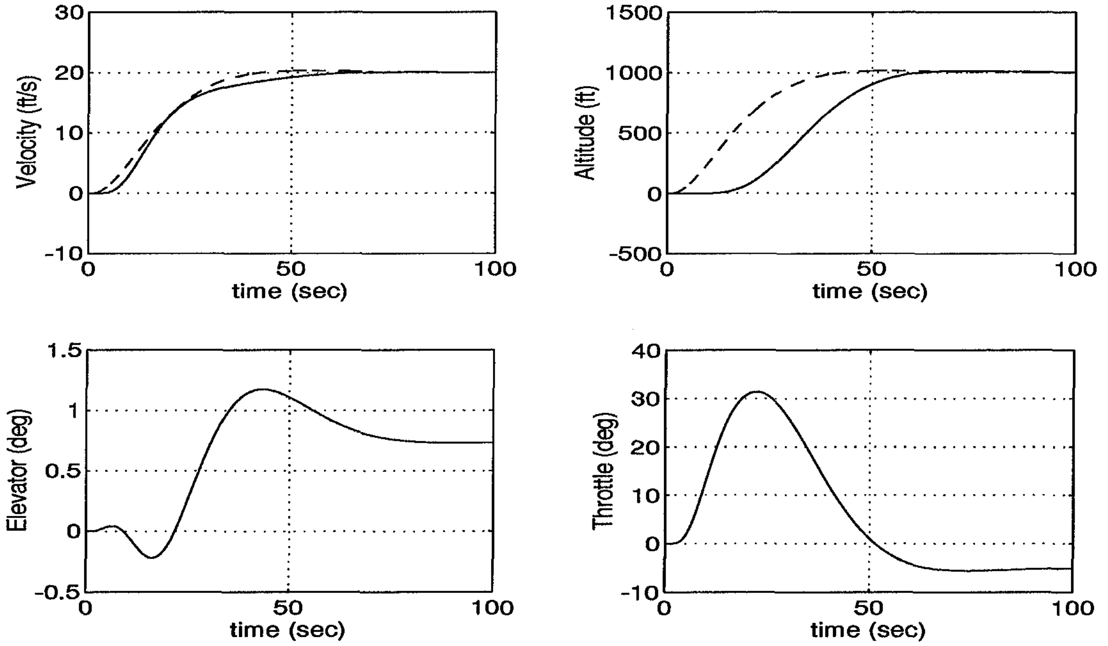


Figure 5.9: Flight Point 2—Linear Aircraft Responses to a 20 ft/s Velocity Command and 1000 ft Altitude Command.

5.2 Nonlinear Closed-Loop Model Evaluation

Using the same TECS controller gains and the system structure as Section 5.1, the linearized F-15 dynamics are now replaced by the nonlinear F-15 model. The system responses are again evaluated in the SIMULINK environment. Nonlinear system responses for the velocity V, the altitude h, the elevator δ_H and the throttle δ_{PLA} are shown below and compared with the equivalent linearized responses from the previous section. The complete set of nonlinear responses is shown in Appendix D.

5.2.1 Command Responses

Velocity Command

Nonlinear aircraft responses to a 20 ft/s velocity command are shown for both flight conditions in Figures 5.10-5.11 . Several differences exist in these responses from the linearized equivalents in Figures 5.4-5.5 . Both flight points show velocity responses with settling times greater than 50 seconds and with some overshoot. The altitude responses exhibit steady-state errors of approximately 40 ft and 50 ft for flight points 1 and 2 respectively. Most notable, however, are the high-frequency oscillations in elevator during the first 40 seconds of the response.

Altitude Command

Nonlinear aircraft responses to a 1000 ft altitude command are shown for both flight conditions in Figures 5.12-5.13 . Here, the differences between the nonlinear and linearized responses are magnified even further. The altitude tracking is similar to that of the linear model until the command value is reached—the nonlinear model exhibits a steady-state error of approximately 50 ft at both flight points. The elevator again oscillates at high frequency, in this case throughout the entire command response. The throttle responses differ significantly from the linearized responses. A much larger steady-state value exists—approximately 18° power level angle for both flight conditions compared to 2° and less than 1° for flight points 1 and 2 respectively in the linear model responses.

Combined Velocity and Altitude Commands

The linearized aircraft responses to simultaneous commands of 20 ft/s in velocity and 1000 ft in altitude are shown for both flight conditions in Figures 5.14-5.15. Of the three sets of nonlinear responses, the combined command results in the greatest similarity to the linear model responses. Again, the elevator response exhibits high-frequency oscillations, but considerably less than the altitude command alone. The altitude tracking also has a steady-state error which is less than that of the altitude command alone. The throttle response, though jerky, corresponds more to the linearized throttle response than either of the previous command responses.

5.0.1 Analysis

The objective of designing a satisfactory TECS controller using a linearized model derived from the overall nonlinear system has been achieved. The implementation of the TECS controller with the nonlinear system, however, uncovered several shortcomings. The high-frequency oscillations in the elevator responses and the irregularities in the throttle responses are most noteworthy. Possible explanations abound, including the possibility of numerical errors in the simulation. Time steps as small as 0.001 seconds were used in an effort to alleviate the high-frequency characteristics.

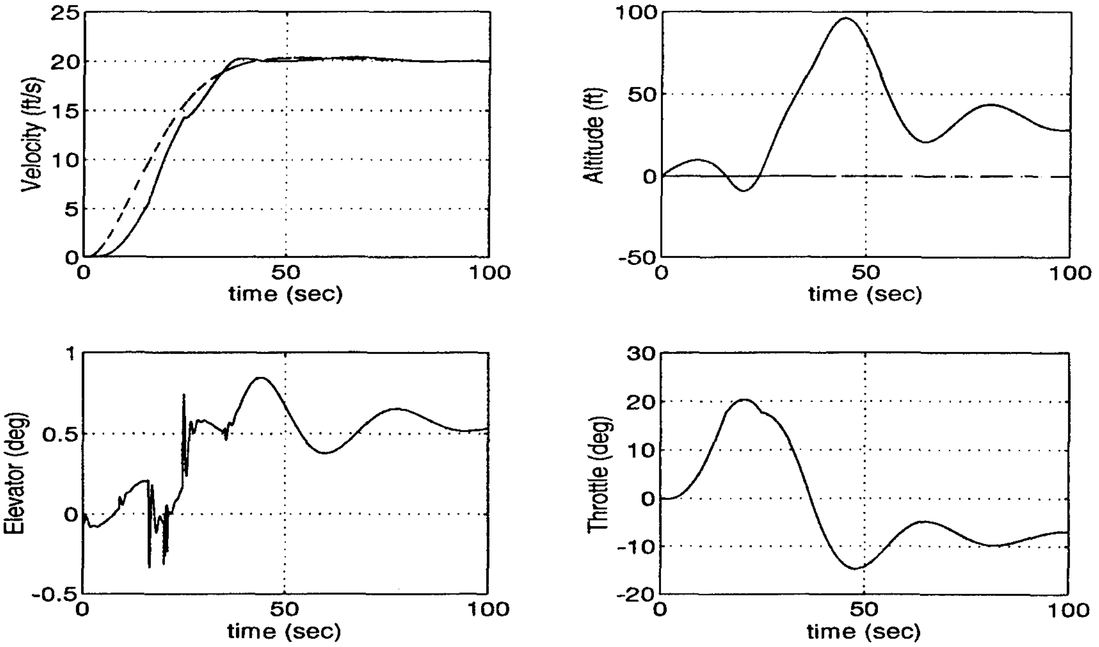


Figure 5.10: Flight Point 1—Nonlinear Aircraft Responses to a 20 ft/s Velocity Command.

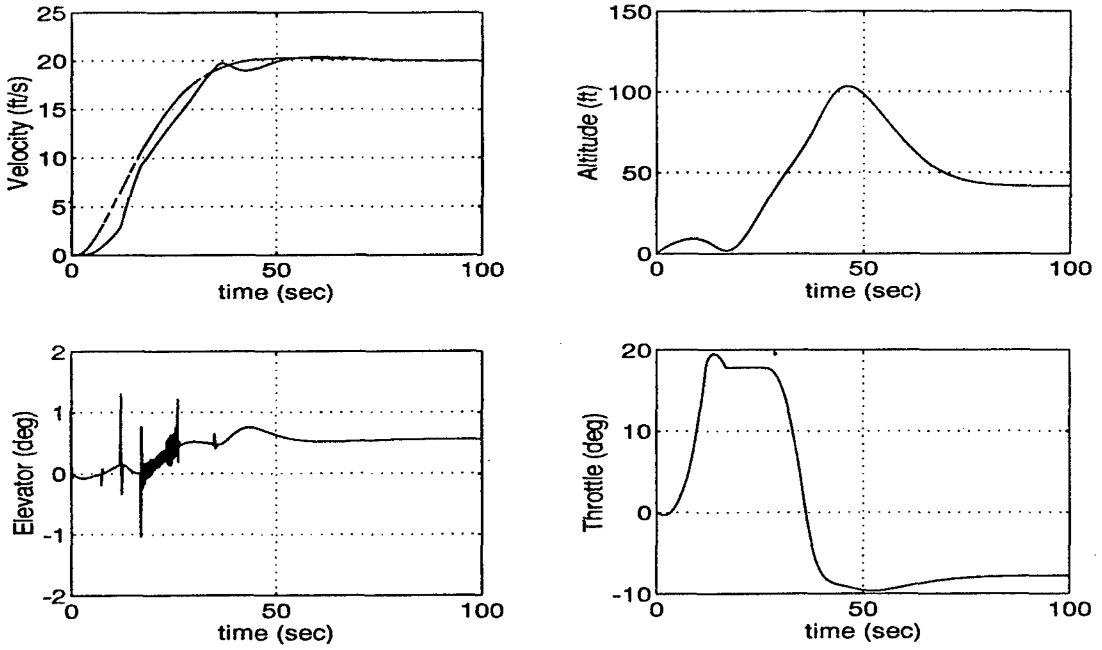


Figure 5.11: Flight Point 2—Nonlinear Aircraft Responses to a 20 ft/s Velocity Command.

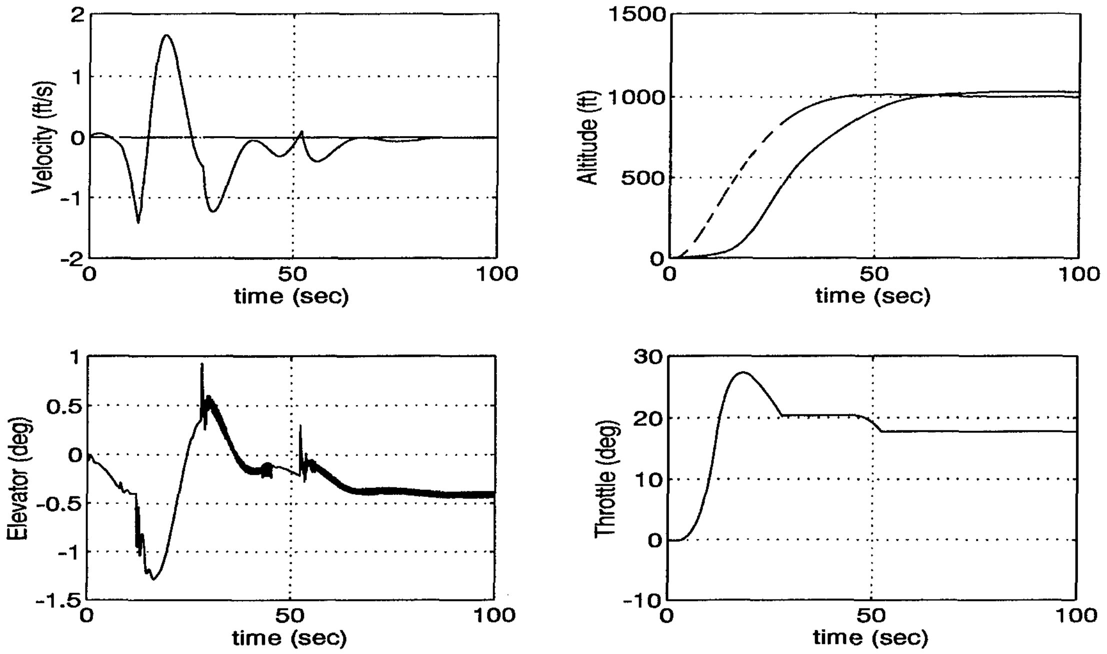


Figure 5.12: Flight Point 1—Nonlinear Aircraft Responses to a 1000 ft Altitude Command.

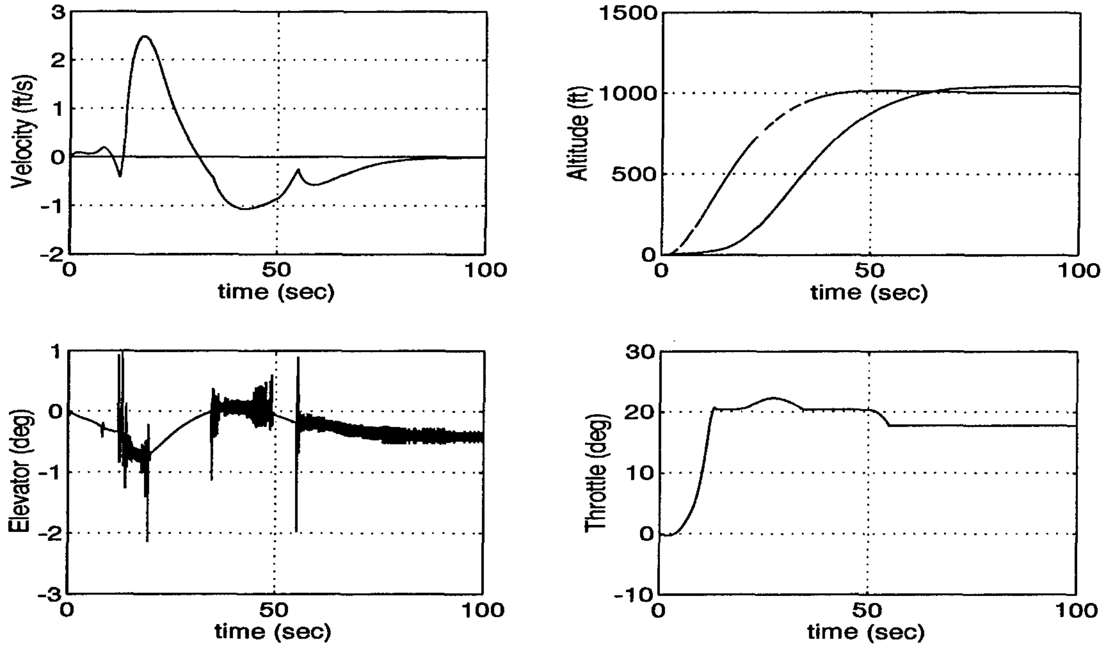


Figure 5.13: Flight Point 2—Nonlinear Aircraft Responses to a 1000 ft Altitude Command.

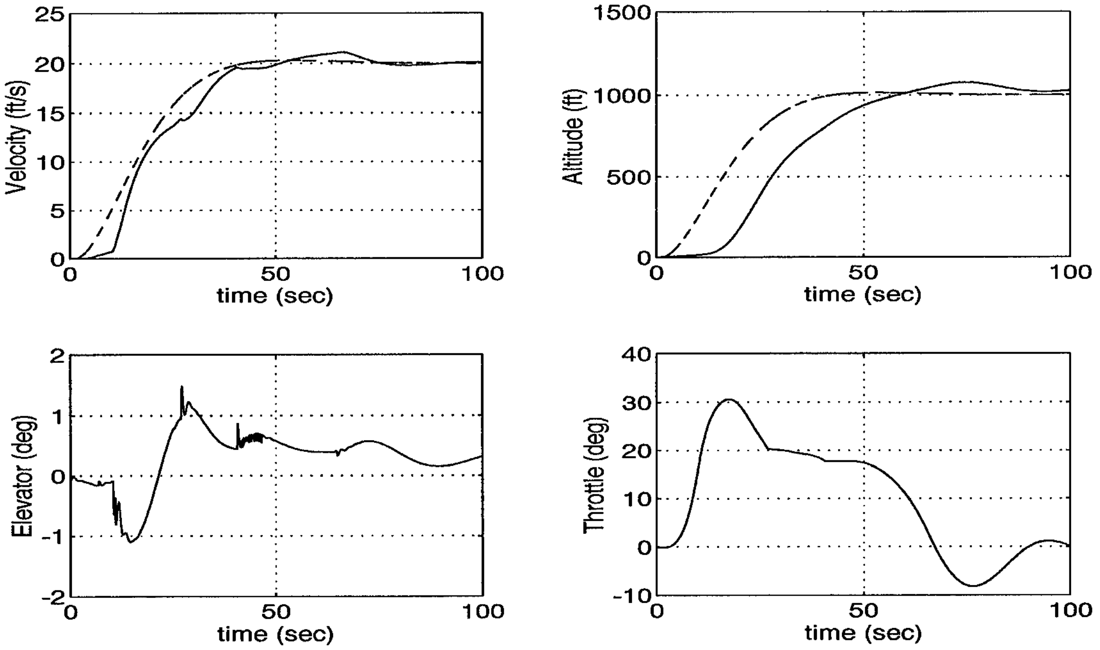


Figure 5.14: Flight Point 1—Nonlinear Aircraft Responses to a 20 ft/s Velocity Command and 1000 ft Altitude Command.

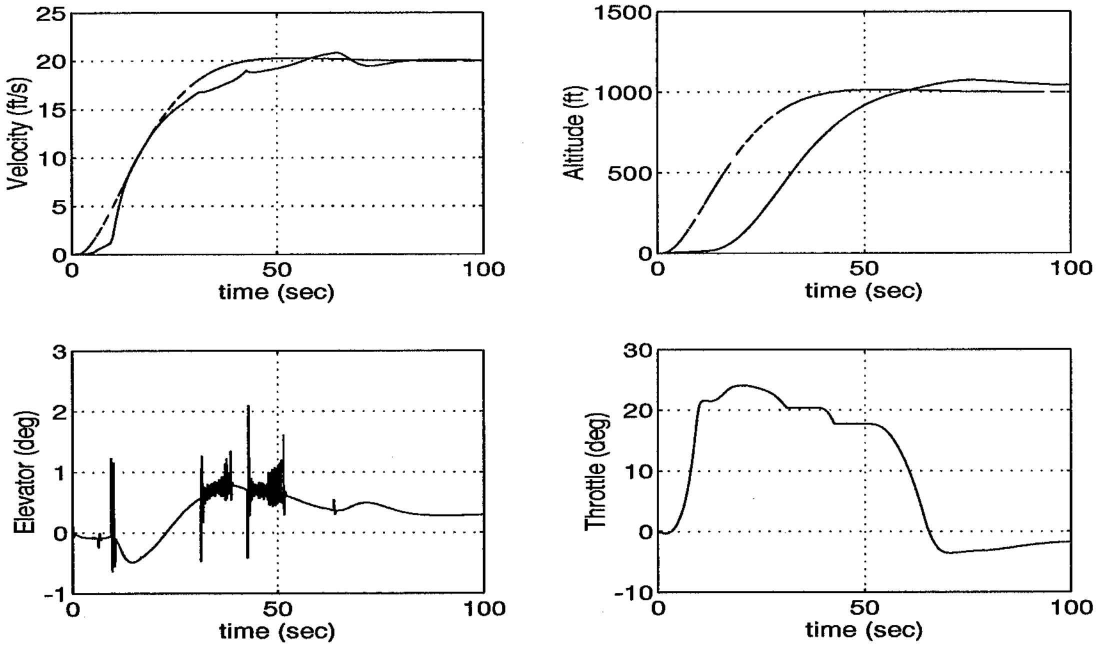


Figure 5.15: Flight Point 2—Nonlinear Aircraft Responses to a 20 ft/s Velocity Command and 1000 ft Altitude Command.

Chapter 6

CONCLUSIONS

6.1 Summary

The objective of this report is to develop a nonlinear aircraft simulation for the F-15, use a subsequent linearized model to develop a control law, and then validate the final controller on the nonlinear model. This model is based on the nonlinear, six-degree-of-freedom equations of motion, with internal modules representing the nonlinear aerodynamic and propulsion characteristics of the aircraft and the atmospheric data.

In validating the nonlinear model at equilibrium conditions, several shortcomings were found in the MATLAB `trim` function. Nevertheless, using the data provided from the Genesis simulation as an initial guess, adequate trim conditions were determined. Linearization of the model at the two selected flight conditions was successfully validated by the comparison of the pulse responses for the linear and nonlinear models. Finally, the longitudinal TECS control law was designed for the system using the linearized dynamics. Although not ideal, the closed-loop characteristics of the linearized model were certainly adequate—satisfactory closed-loop stability, robustness, turbulence rejection, and control bandwidths were all verified. The linearized command responses exhibited adequate settling times and good tracking characteristics, with zero steady-state error.

Implementation of the TECS controller on the nonlinear model was successful with respect to achieving stable command responses similar to those of the linearized model responses. However, several differences were identified which could not be readily explained. They include high-frequency elevator-response characteristics, steady-state errors in altitude, irregular throttle-response characteristics and high steady-state throttle values. Since the intent of using the nonlinear model is to validate the controller, these characteristics ideally would be explained by shortcomings in the control law. However, that is not likely the case here. Based on the irregular nature of the variations between the linear and nonlinear model responses, the greatest likelihood is that the interaction between the nonlinear model and the controller is not yet perfected. Future improvements in the nonlinear model will be required to make it a useful tool to the designer for the final validation of a control-law design.

6.2 Recommendations for Future Study

Many of the areas requiring further investigation have already been mentioned. The following list summarizes the primary areas of concern which must be addressed to make the nonlinear model a viable tool to the controls engineer:

- Improve or redesign the MATLAB `trim` function to allow *fixed* values to be set for the states, outputs, or inputs and to allow the function greater flexibility in terms of selecting an initial guess. Also, improve the numerical efficiency of the function to decrease the time required to generate a trim condition.

- Address the problems with the high Mach number flight regime and verify that they are specific to the F-15 model data and not to the generic nonlinear model structure.

- Investigate and determine the causes of the undesireable closed-loop characteristics in the nonlinear model responses mentioned in the previous section. Verify whether these shortcomings result from the aerodynamic and propulsion modules, making them specific to the F-15 data, or if they are the result of shortcomings in the overall nonlinear model structure.

- Apply the nonlinear model framework to other aircraft using their specific nonlinear aerodynamic and propulsion characteristics. Design an appropriate control-law using a linearized model and then validate with the nonlinear model.

The ultimate viability of the nonlinear model will depend on the satisfaction of most, if not all, of the above areas.

BIBLIOGRAPHY

[1] Anderson, John D. Jr., *Introduction to Flight.* McGraw-Hill Book Company, 3rd Edition, 1989.

[2] Bruce, K.R., *Integrated Autopilot/Autothrottle Based on a Total Energy Control Concept: Design and Evaluation of Additional Autopilot Modes.* Technical Report NASA TCV Contract NAS1-16300, Boeing Commercial Airplane Company, August 1987.

[3] Bruce, K.R., *Integrated Autopilot/Autothrottle for the NASA TSRV B-737 Aircraft: Design and Verification by Nonlinear Simulation.* Technical Report NASA CR 4217, NASA Langley Research Center, February 1989.

[4] Bruce, K.R., J.R. Kelly, and L.H. Person, *NASA B737 Flight Test Results of the Total Energy Control System.* Technical Report AIAA 86-2143-CP, AIAA Guidance, Navigation and Control Conference, August 1986.

[5] Brumbaugh, Randal W., "An Aircraft Model for the AIAA Controls Design Challenge," PRC Inc., Edwards, CA.

[6] Duke, E.L., Antoniewicz, R.F., and Krambeer, K.D., *Derivation and Definition of a Linear Aircraft Model*, NASA RP-1207, Aug. 1988.

[7] Lambregts, A.A., *Integrated System Design for Flight and Propulsion Control using Total Energy Principles.* Technical Report AIAA 83-2561, AIAA, October 1983.

[8] Lambregts, A.A., *Operational Aspects of the Integrated Vertical Flight Path and Speed Control System.* Technical Report SAE 831420, Aerospace Congress & Exposition, Long Beach, California, August 1983.

[9] Swamy, Sanjay, "Robust Integrated Autopilot Design Using Constrained Parameter Optimization," University of Washington, Department of Aeronautics and Astronautics, 1992.

Appendix A

F-15 NONLINEAR SIMULATION S-FUNCTIONS

A.1 S-Function for Open-Loop F-15 Model

The following is a listing of the MATLAB s-function f25sfn used for the open-loop nonlinear F-15 simulation.

```
%%% NONLINEAR OPEN-LOOP F-15 SIMULATION S-FUNCTION %%%

function [sys,x0] = f25sfn(t,x,u,flag)

global A IA
global V alpha q theta p phi r psi beta xp yp h
global DH DD DA DR PLAPL PLAPR PLASYM alpdot
global xx utrim

%%% UPDATE A,IA-ARRAYS %%%
A(829)  = V;       A(914)  = alpha;  A(862)  = q;
A(713)  = h;       A(715)  = xp;     A(716)  = yp;
A(943)  = theta;   A(861)  = p;      A(942)  = phi;
A(863)  = r;       A(944)  = psi;    A(915)  = beta;
A(1402) = DH;      A(1403) = DD;     A(1401) = DA;
A(1404) = DR;      A(1416) = PLAPL;  A(1417) = PLAPR;
A(1418) = PLASYM;

%%% STANDARD ATMOSPHERE %%%
[AMCH,RHO,QBAR,G] = atmos(A);

%%% A-ARRAY UPDATE %%%
A(825) = AMCH;     A(670) = RHO;
```

```
A(669) = QBAR;    A(772) = G;

%%% STABILITY AXIS FORCES AND MOMENTS %%%
[CLFT,CD,CY,CL,CM,CN,FAX,FAY,FAZ,ALM,AMM,ANM] = f25aero(A,IA);

%% TRANSFER TO A-ARRAY
A(1410) = CLFT;   A(1411) = CD;     A(1412) = CY;
A(1413) = CL;     A(1414) = CM;     A(1415) = CN;
A(748)  = FAX;    A(749)  = FAY;    A(750)  = FAZ;
A(733)  = ALM;    A(734)  = AMM;    A(735)  = ANM;

%%% CALCULATE PROPULSION FORCES %%%
[FPX,FPY,FPZ,DCL,DCM,DCN,TAUL,TAUR,PLAL,PLAR,COUT1C,COUT2C,...
 FIRST] = f25eng(A,IA);

%%% TRANSFER TO THE A AND IA ARRAYS %%%
A(751)  = FPX;     A(752)  = FPY;     A(753) = FPZ;
A(736)  = DCL;     A(737)  = DCM;     A(738) = DCN;
A(1419) = TAUL;    A(1420) = TAUR;
A(1431) = PLAL;    A(1432) = PLAR;
A(667)  = COUT1C;  A(668)  = COUT2C;
IA(502) = FIRST;

%%% A,IA-ARRAY VAR NAMES %%%
S    = A(659);    QBAR = A(669);    g    = A(772);
CLFT = A(1410);   CD   = A(1411);   CY   = A(1412);
ALM  = A(733);    AMM  = A(734);    ANM  = A(735);
FPX  = A(751);    FPY  = A(752);    FPZ  = A(753);
Ix   = A(634);    Iy   = A(635);    Iz   = A(636);
Ixz  = A(637);    Ixy  = A(638);    Iyz  = A(639);
I1   = A(632);    I2   = A(631);    I3   = A(630);
I4   = A(629);    I5   = A(628);    I6   = A(627);
Dx   = A(626);    Dy   = A(625);    Dz   = A(624);
```

```
detI = A(633);    m     = A(658)/G;

%%% CALCULATE UPDATED AERO VALUES %%%
L = QBAR*S*CLFT;
D = QBAR*S*CD;
Y = QBAR*S*CY;
SumL = ALM;
SumM = AMM;
SumN = ANM;
XT = FPX;
YT = FPY;
ZT = FPZ;

%%% ANGLE CALCULATIONS %%%
COSTHETA = cos(theta);
SINTHETA = sin(theta);
TANTHETA = tan(theta);
SECTHETA = 1/cos(theta);
COSBETA  = cos(beta);
SINBETA  = sin(beta);
TANBETA  = tan(beta);
COSALPHA = cos(alpha);
SINALPHA = sin(alpha);
COSPHI   = cos(phi);
SINPHI   = sin(phi);
COSPSI   = cos(psi);
SINPSI   = sin(psi);

%%% EQUATIONS OF MOTION %%%
V1 = -D*COSBETA+Y*SINBETA+XT*COSALPHA*COSBETA;
V2 = YT*SINBETA+ZT*SINALPHA*COSBETA;
V3 = -m*g*(COSALPHA*COSBETA*SINTHETA);
V4 = -m*g*(-SINBETA*SINPHI*COSTHETA);
```

```
V5 = -m*g*(-SINALPHA*COSBETA*COSPHI*COSTHETA);
al1 = -L+ZT*COSALPHA-XT*SINALPHA;
al2 = m*g*(COSALPHA*COSPHI*COSTHETA+SINALPHA*SINTHETA);
al3 = q-TANBETA*(p*COSALPHA+r*SINALPHA);
q1 = SumL*I2+SumM*I4+SumN*I5-p^2*(Ixz*I4-Ixy*I5);
q2 = p*q*(Ixz*I2-Iyz*I4-Dz*I5)-p*r*(Ixy*I2+Dy*I4-Iyz*I5);
q3 = q^2*(Iyz*I2-Ixy*I5)-q*r*(Dx*I2-Ixy*I4+Ixz*I5);
q4 = -r^2*(Iyz*I2-Ixz*I4);
p1 = SumL*I1+SumM*I2+SumN*I3-p^2*(Ixz*I2-Ixy*I3);
p2 = p*q*(Ixz*I1-Iyz*I2-Dz*I3)-p*r*(Ixy*I1+Dy*I2-Iyz*I3);
p3 = q^2*(Iyz*I1-Ixy*I3)-q*r*(Dx*I1-Ixy*I2+Ixz*I3);
p4 = -r^2*(Iyz*I1-Ixz*I2);
r1 = SumL*I3+SumM*I5+SumN*I6-p^2*(Ixz*I5-Ixy*I6);
r2 = p*q*(Ixz*I3-Iyz*I5-Dz*I6)-p*r*(Ixy*I3+Dy*I5-Iyz*I6);
r3 = q^2*(Iyz*I3-Ixy*I6)-q*r*(Dx*I3-Ixy*I5+Ixz*I6);
r4 = -r^2*(Iyz*I3-Ixz*I5);
be1 = D*SINBETA+Y*COSBETA-XT*COSALPHA*SINBETA;
be2 = YT*COSBETA-ZT*SINALPHA*SINBETA;
be3 = m*g*(COSALPHA*SINBETA*SINTHETA);
be4 = m*g*(COSBETA*SINPHI*COSTHETA);
be5 = m*g*(-SINALPHA*SINBETA*COSPHI*COSTHETA);
be6 = p*SINALPHA-r*COSALPHA;
xp1 = COSALPHA*COSBETA*COSTHETA*COSPSI;
xp2 = SINBETA*(SINPHI*SINTHETA*COSPSI-COSPHI*SINPSI);
xp3 = SINALPHA*COSBETA*(COSPHI*SINTHETA*SINPSI-SINPHI*COSPSI);
yp1 = COSALPHA*COSBETA*COSTHETA*SINPSI;
yp2 = SINBETA*(COSPHI*COSPSI+SINPHI*SINTHETA*SINPSI);
yp3 = SINALPHA*COSBETA*(COSPHI*SINTHETA*SINPSI-SINPHI*COSPSI);
h1 = COSALPHA*COSBETA*SINTHETA;
h2 = SINBETA*SINPHI*COSTHETA;
h3 = SINALPHA*COSBETA*COSPHI*COSTHETA;

if flag == 0
```

```
  %%% SYSTEM CHARACTERISTICS/INITIAL CONDITIONS %%%
  sys = [12 0 13 8 12 0];
  x0  = xx;

  %%% STATES (X) %%%
  %%% LONGITUDINAL STATES %%%
  V          = x0(1);   %%% TOTAL VEHICLE VELOCITY (FT/S) %%%
  alpha      = x0(2);   %%% ANGLE OF ATTACK (RAD)         %%%
  q          = x0(3);   %%% PITCH RATE (RAD/S)            %%%
  theta      = x0(4);   %%% PITCH ANGLE (RAD)             %%%
  %%% LATERAL/DIRECTIONAL STATES %%%
  p          = x0(5);   %%% ROLL RATE (RAD/S)             %%%
  phi        = x0(6);   %%% ROLL ANGLE (RAD)              %%%
  r          = x0(7);   %%% YAW RATE (RAD/S)              %%%
  psi        = x0(8);   %%% YAW ANGLE (RAD)               %%%
  beta       = x0(9);   %%% SIDESLIP ANGLE (RAD)          %%%
  %%% EARTH-RELATIVE POSITION STATES %%%
  xp         = x0(10);  %%% X-DIRECTION POSITION (FT)     %%%
  yp         = x0(11);  %%% Y-DIRECTION POSITION (FT)     %%%
  h          = x0(12);  %%% ALTITUDE (FT)                 %%%

elseif abs(flag) == 1
  %%% STATES (X) %%%
  %%% LONGITUDINAL STATES %%%
  V          = x(1);    %%% TOTAL VEHICLE VELOCITY (FT/S) %%%
  alpha      = x(2);    %%% ANGLE OF ATTACK (RAD)         %%%
  q          = x(3);    %%% PITCH RATE (RAD/S)            %%%
  theta      = x(4);    %%% PITCH ANGLE (RAD)             %%%
  %%% LATERAL/DIRECTIONAL STATES %%%
  p          = x(5);    %%% ROLL RATE (RAD/S)             %%%
  phi        = x(6);    %%% ROLL ANGLE (RAD)              %%%
  r          = x(7);    %%% YAW RATE (RAD/S)              %%%
  psi        = x(8);    %%% YAW ANGLE (RAD)               %%%
  beta      = x(9);    %%% SIDESLIP ANGLE (RAD)           %%%
  %%% EARTH-RELATIVE POSITION STATES %%%
  xp        = x(10);   %%% X-DIRECTION POSITION (FT)      %%%
  yp        = x(11);   %%% Y-DIRECTION POSITION (FT)      %%%
  h         = x(12);   %%% ALTITUDE (FT)                  %%%

  %%% INPUTS (U) %%%
  DH        = u(1)+utrim(1);    %%% SYMETRIC STABILATOR (DEG)      %%%
  DD        = u(2);             %%% DIFFERENTIAL STABILATOR (DEG) %%%
  DA        = u(3)+utrim(5);    %%% AILERON DEFLECTION (DEG)       %%%
  DR        = u(4)+utrim(6);    %%% RUDDER DEFLECTION (DEG)        %%%
  PLAPL     = u(5)+utrim(2);    %%% LEFT  PLA (DEG)                %%%
  PLAPR     = u(6)+utrim(3);    %%% RIGHT PLA (DEG)                %%%
  PLASYM    = u(7);             %%% SYMMETRIC PLA (DEG)            %%%
  alpdot    = u(8);             %%% AOA RATE (RAD/S)               %%%

  %%% STATE DERIVATIVES (dX/dT) %%%
  sys(1,1) = (V1+V2+V3+V4+V5)/m;
  sys(2,1) = al3+(al1+al2)/(V*m*COSBETA);
  sys(3,1) = (q1+q2+q3+q4)/detI;
  sys(4,1) = q*COSPHI-r*SINPHI;
  sys(5,1) = (p1+p2+p3+p4)/detI;
  sys(6,1) = p+q*SINPHI*TANTHETA+r*COSPHI*TANTHETA;
  sys(7,1) = (r1+r2+r3+r4)/detI;
  sys(8,1) = r*COSPHI*SECTHETA+q*SINPHI*SECTHETA;
  sys(9,1) = be6+(be1+be2+be3+be4+be5)/(m*V);
  sys(10,1) = V*(xp1+xp2+xp3);
  sys(11,1) = V*(yp1+yp2+yp3);
  sys(12,1) = V*(h1-h2-h3);

elseif flag == 3
  %%% SYSTEM OUTPUTS (Y) %%%
  sys(1,1) = x(1);
  sys(2,1) = x(2);
  sys(3,1) = x(3);
  sys(4,1) = x(4);
  sys(5,1) = x(5);
  sys(6,1) = x(6);
  sys(7,1) = x(7);
  sys(8,1) = x(8);
  sys(9,1) = x(9);
  sys(10,1) = al3+(al1+al2)/(V*m*COSBETA);
  sys(11,1) = x(10);
  sys(12,1) = x(11);
  sys(13,1) = x(12);

  else
    %%% ALL OTHER FLAGS UNDECLARED %%%
    sys = [];
  end
end
```

A.2 S-Function for Closed-Loop F-15 Model

The following is a listing of the MATLAB s-function `f25sfncl` used for the closed-loop nonlinear F-15 simulation:

```
%%% NONLINEAR CLOSED-LOOP F-15 SIMULATION S-FUNCTION %%%

function [sys,x0] = f25sfncl(t,x,u,flag)

global A IA
global V alpha q theta p phi r psi beta xp yp h
global DH  PLAPL PLAPR FLAPS DA DR PLASYM DD
global xx utrim

%%% UPDATE A,IA-ARRAYS %%%
```

```
A(829)  = V;      A(914)  = alpha; A(862)  = q;
A(713)  = h;      A(715)  = xp;    A(716)  = yp;
A(943)  = theta;  A(861)  = p;     A(942)  = phi;
A(863)  = r;      A(944)  = psi;   A(915)  = beta;
A(1402) = DH;     A(1403) = DD;    A(1401) = DA;
A(1404) = DR;     A(1416) = PLAPL; A(1417) = PLAPR;
A(1418) = PLASYM;

%%% STANDARD ATMOSPHERE %%%
[AMCH,RHO,QBAR,G] = atmos(A);

%%% A-ARRAY UPDATE %%%
A(825) = AMCH;    A(670) = RHO;
A(669) = QBAR;    A(772) = G;

%%% STABILITY AXIS FORCES AND MOMENTS %%%
[CLFT,CD,CY,CL,CM,CN,FAX,FAY,FAZ,ALM,AMM,ANM] = f25aero(A,IA);

%% TRANSFER TO THE A-ARRAY %%%
A(1410) = CLFT;   A(1411) = CD;    A(1412) = CY;
A(1413) = CL;     A(1414) = CM;    A(1415) = CN;
A(748)  = FAX;    A(749)  = FAY;   A(750)  = FAZ;
A(733)  = ALM;    A(734)  = AMM;   A(735)  = ANM;

%%% CALCULATE PROPULSION FORCES %%%
[FPX,FPY,FPZ,DCL,DCM,DCN,TAUL,TAUR,PLAL,PLAR,COUT1C,COUT2C,...
 FIRST] = f25eng(A,IA);

%% TRANSFER TO THE A AND IA ARRAYS %%%
A(751)  = FPX;     A(752)  = FPY;     A(753) = FPZ;
A(736)  = DCL;     A(737)  = DCM;     A(738) = DCN;
A(1419) = TAUL;    A(1420) = TAUR;
A(1431) = PLAL;    A(1432) = PLAR;
```

```
A(667)  = COUT1C;  A(668)  = COUT2C;
IA(502) = FIRST;

%%% A,IA-ARRAY VAR NAMES %%%
S    = A(659);     QBAR = A(669);     g    = A(772);
CLFT = A(1410);    CD   = A(1411);    CY   = A(1412);
ALM  = A(733);     AMM  = A(734);     ANM  = A(735);
FPX  = A(751);     FPY  = A(752);     FPZ  = A(753);
Ix   = A(634);     Iy   = A(635);     Iz   = A(636);
Ixz  = A(637);     Ixy  = A(638);     Iyz  = A(639);
I1   = A(632);     I2   = A(631);     I3   = A(630);
I4   = A(629);     I5   = A(628);     I6   = A(627);
Dx   = A(626);     Dy   = A(625);     Dz   = A(624);
detI = A(633);     m    = A(658)/G;

%%% CALCULATE UPDATED AERO VALUES %%%
L = QBAR*S*CLFT;
D = QBAR*S*CD;
Y = QBAR*S*CY;
SumL = ALM;
SumM = AMM;
SumN = ANM;
XT = FPX;
YT = FPY;
ZT = FPZ;

%%% ANGLE CALCULATIONS %%%
COSTHETA = cos(theta);
SINTHETA = sin(theta);
TANTHETA = tan(theta);
SECTHETA = 1/cos(theta);
COSBETA  = cos(beta);
SINBETA  = sin(beta);
```

```
TANBETA  = tan(beta);
COSALPHA = cos(alpha);
SINALPHA = sin(alpha);
COSPHI   = cos(phi);
SINPHI   = sin(phi);
COSPSI   = cos(psi);
SINPSI   = sin(psi);

%%% EQUATIONS OF MOTION %%%
V1 = -D*COSBETA+Y*SINBETA+XT*COSALPHA*COSBETA;
V2 = YT*SINBETA+ZT*SINALPHA*COSBETA;
V3 = -m*g*(COSALPHA*COSBETA*SINTHETA);
V4 = -m*g*(-SINBETA*SINPHI*COSTHETA);
V5 = -m*g*(-SINALPHA*COSBETA*COSPHI*COSTHETA);
al1 = -L+ZT*COSALPHA-XT*SINALPHA;
al2 = m*g*(COSALPHA*COSPHI*COSTHETA+SINALPHA*SINTHETA);
al3 = q-TANBETA*(p*COSALPHA+r*SINALPHA);
q1 = SumL*I2+SumM*I4+SumN*I5-p^2*(Ixz*I4-Ixy*I5);
q2 = p*q*(Ixz*I2-Iyz*I4-Dz*I5)-p*r*(Ixy*I2+Dy*I4-Iyz*I5);
q3 = q^2*(Iyz*I2-Ixy*I5)-q*r*(Dx*I2-Ixy*I4+Ixz*I5);
q4 = -r^2*(Iyz*I2-Ixz*I4);
p1 = SumL*I1+SumM*I2+SumN*I3-p^2*(Ixz*I2-Ixy*I3);
p2 = p*q*(Ixz*I1-Iyz*I2-Dz*I3)-p*r*(Ixy*I1+Dy*I2-Iyz*I3);
p3 = q^2*(Iyz*I1-Ixy*I3)-q*r*(Dx*I1-Ixy*I2+Ixz*I3);
p4 = -r^2*(Iyz*I1-Ixz*I2);
r1 = SumL*I3+SumM*I5+SumN*I6-p^2*(Ixz*I5-Ixy*I6);
r2 = p*q*(Ixz*I3-Iyz*I5-Dz*I6)-p*r*(Ixy*I3+Dy*I5-Iyz*I6);
r3 = q^2*(Iyz*I3-Ixy*I6)-q*r*(Dx*I3-Ixy*I5+Ixz*I6);
r4 = -r^2*(Iyz*I3-Ixz*I5);
be1 = D*SINBETA+Y*COSBETA-XT*COSALPHA*SINBETA;
be2 = YT*COSBETA-ZT*SINALPHA*SINBETA;
be3 = m*g*(COSALPHA*SINBETA*SINTHETA);
be4 = m*g*(COSBETA*SINPHI*COSTHETA);
```

```
be5 = m*g*(-SINALPHA*SINBETA*COSPHI*COSTHETA);
be6 = p*SINALPHA-r*COSALPHA;
xp1 = COSALPHA*COSBETA*COSTHETA*COSPSI;
xp2 = SINBETA*(SINPHI*SINTHETA*COSPSI-COSPHI*SINPSI);
xp3 = SINALPHA*COSBETA*(COSPHI*SINTHETA*SINPSI-SINPHI*COSPSI);
yp1 = COSALPHA*COSBETA*COSTHETA*SINPSI;
yp2 = SINBETA*(COSPHI*COSPSI+SINPHI*SINTHETA*SINPSI);
yp3 = SINALPHA*COSBETA*(COSPHI*SINTHETA*SINPSI-SINPHI*COSPSI);
h1 = COSALPHA*COSBETA*SINTHETA;
h2 = SINBETA*SINPHI*COSTHETA;
h3 = SINALPHA*COSBETA*COSPHI*COSTHETA;

if flag == 0
  %%% SYSTEM CHARACTERISTICS/INITIAL CONDITIONS %%%
  sys = [12 0 14 6 12 0];
  x0  = xx;

  %%% STATES (X) %%%
  %%% LONGITUDINAL STATES %%%
  V         = x0(1);   %%% TOTAL VEHICLE VELOCITY (FT/S)    %%%
  alpha     = x0(2);   %%% ANGLE OF ATTACK (RAD)            %%%
  q         = x0(3);   %%% PITCH RATE (RAD/S)               %%%
  theta     = x0(4);   %%% PITCH ANGLE (RAD)                %%%
  %%% LATERAL/DIRECTIONAL STATES %%%
  p         = x0(5);   %%% ROLL RATE (RAD/S)                %%%
  phi       = x0(6);   %%% ROLL ANGLE (RAD)                 %%%
  r         = x0(7);   %%% YAW RATE (RAD/S)                 %%%
  psi       = x0(8);   %%% YAW ANGLE (RAD)                  %%%
  beta      = x0(9);   %%% SIDESLIP ANGLE (RAD)             %%%
  %%% EARTH-RELATIVE POSITION STATES %%%
  xp        = x0(10);  %%% X-DIRECTION POSITION (FT)        %%%
  yp        = x0(11);  %%% Y-DIRECTION POSITION (FT)        %%%
  h         = x0(12);  %%% ALTITUDE (FT)                    %%%
```

```
%%% LONGITUDINAL STATES %%%
V         = x(1);     %%% TOTAL VEHICLE VELOCITY (FT/S)    %%%
alpha     = x(2);     %%% ANGLE OF ATTACK (RAD)            %%%
q         = x(3);     %%% PITCH RATE (RAD/S)               %%%
theta     = x(4);     %%% PITCH ANGLE (RAD)                %%%
%%% LATERAL/DIRECTIONAL STATES %%%
p         = x(5);     %%% ROLL RATE (RAD/S)                %%%
phi       = x(6);     %%% ROLL ANGLE (RAD)                 %%%
r         = x(7);     %%% YAW RATE (RAD/S)                 %%%
psi       = x(8);     %%% YAW ANGLE (RAD)                  %%%
beta      = x(9);     %%% SIDESLIP ANGLE (RAD)             %%%
%%% EARTH-RELATIVE POSITION STATES %%%
xp        = x(10);    %%% X-DIRECTION POSITION (FT)        %%%
yp        = x(11);    %%% Y-DIRECTION POSITION (FT)        %%%
h         = x(12);    %%% ALTITUDE (FT)                    %%%

%%% INPUTS (U) %%%
DH        = u(1)+utrim(1);  %%% SYMETRIC STABILATOR (DEG) %%%
PLAPL     = u(2)+utrim(2);  %%% LEFT  PLA (DEG)           %%%
PLAPR     = u(3)+utrim(3);  %%% RIGHT PLA (DEG)           %%%
FLAPS     = u(4)+utrim(4);  %%% FLAPS (DEG)               %%%
DA        = u(5)+utrim(5);  %%% AILERON DEFLECTION (DEG)  %%%
DR        = u(6)+utrim(6);  %%% RUDDER DEFLECTION (DEG)   %%%
PLASYM    = 0;
DD        = 0;

%%% STATE DERIVATIVES (dX/dT) %%%
sys(1,1) = (V1+V2+V3+V4+V5)/m;
sys(2,1) = al3+(al1+al2)/(V*m*COSBETA);
sys(3,1) = (q1+q2+q3+q4)/detI;
sys(4,1) = q*COSPHI-r*SINPHI;
sys(5,1) = (p1+p2+p3+p4)/detI;
```

```
  sys(6,1) = p+q*SINPHI*TANTHETA+r*COSPHI*TANTHETA;
  sys(7,1) = (r1+r2+r3+r4)/detI;
  sys(8,1) = r*COSPHI*SECTHETA+q*SINPHI*SECTHETA;
  sys(9,1) = be6+(be1+be2+be3+be4+be5)/(m*V);
  sys(10,1) = V*(xp1+xp2+xp3);
  sys(11,1) = V*(yp1+yp2+yp3);
  sys(12,1) = V*(h1-h2-h3);

elseif flag == 3
  %%% SYSTEM OUTPUTS (Y) %%%
  sys(1,1)   = (V1+V2+V3+V4+V5)/m/g;
  sys(2,1)   = x(1)-xx(1);
  sys(3,1)   = x(12)-xx(12);
  sys(4,1)   = x(4)-xx(4)-x(2)+xx(2);
  sys(5,1)   = x(3)-xx(3);
  sys(6,1)   = x(4)-xx(4);
  sys(7,1)   = x(2)-xx(2);
  sys(8,1)   = x(5)-xx(5);
  sys(9,1)   = x(6)-xx(6);
  sys(10,1)  = be6+(be1+be2+be3+be4+be5)/(m*V);
  sys(11,1) = x(9)-xx(9);
  sys(12,1) = x(8)-xx(8);
  sys(13,1) = r*COSPHI*SECTHETA+q*SINPHI*SECTHETA;
  sys(14,1) = x(7)-xx(7);

  else
    %%% ALL OTHER FLAGS UNDECLARED %%%
    sys = [];
  end
end
```

Appendix B

NONLINEAR MODEL RESPONSES AT EQUILIBRIUM

Aircraft responses at the equilibrium point obtained from the Genesis and MATLAB `trim` functions are shown in Figures B.1-B.4 for both flight conditions.

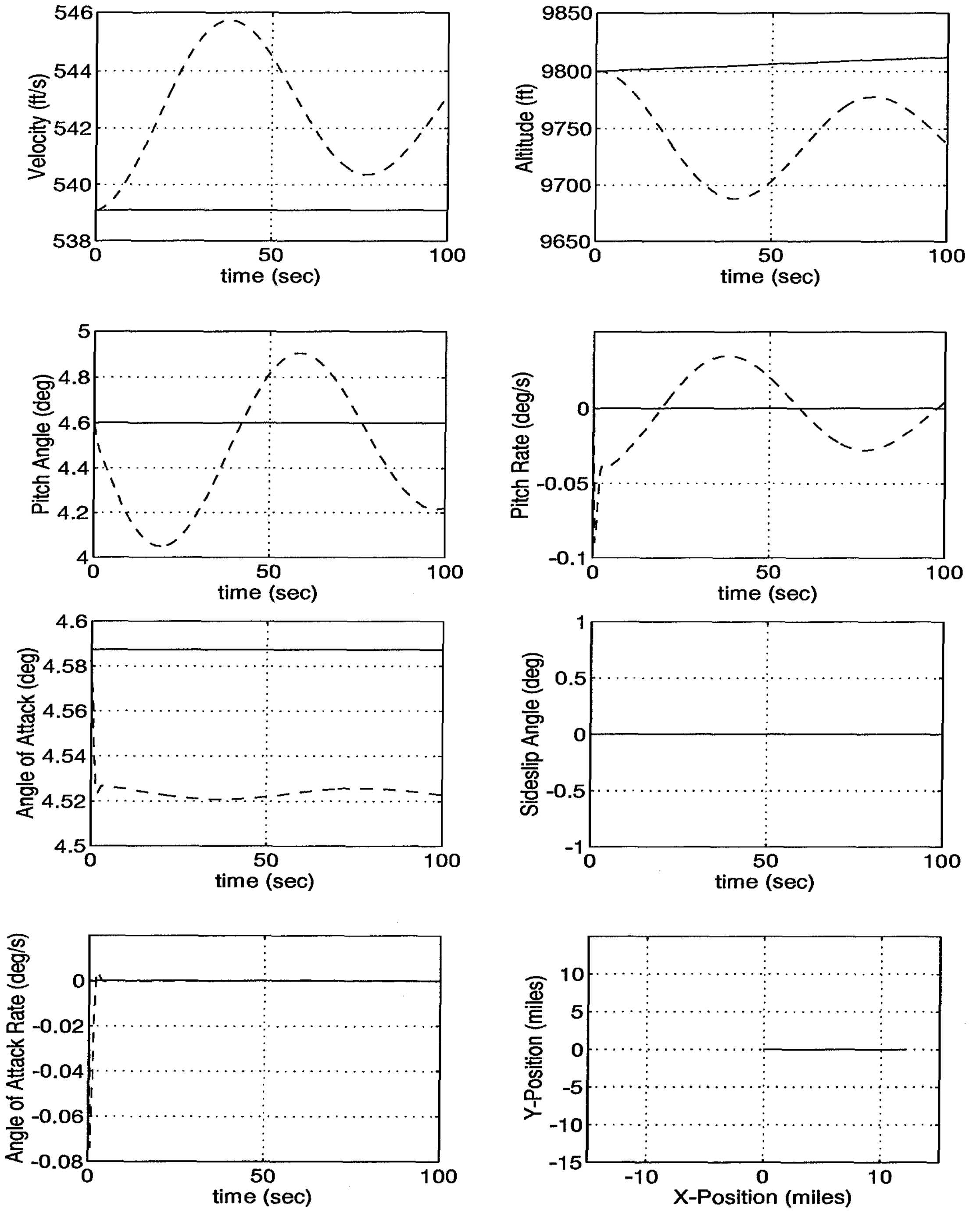


Figure B.1: Flight Point 1—Aircraft Responses to Initial Conditions Set at Trim Values: MATLAB (solid line) and Genesis (dashed line).

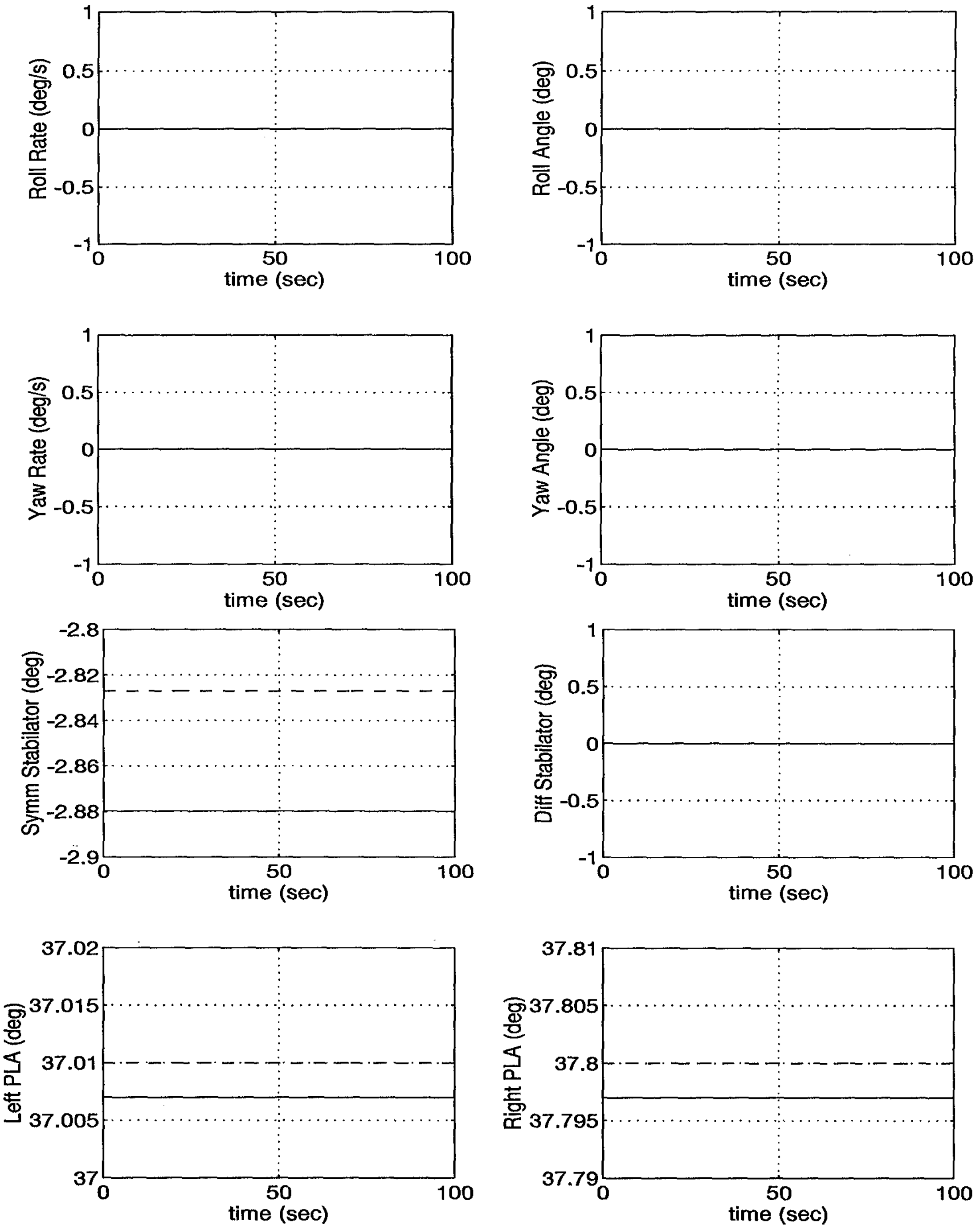


Figure B.2: Flight Point 1—Aircraft Responses to Initial Conditions Set at Trim Values: MATLAB (solid line) and Genesis (dashed line).

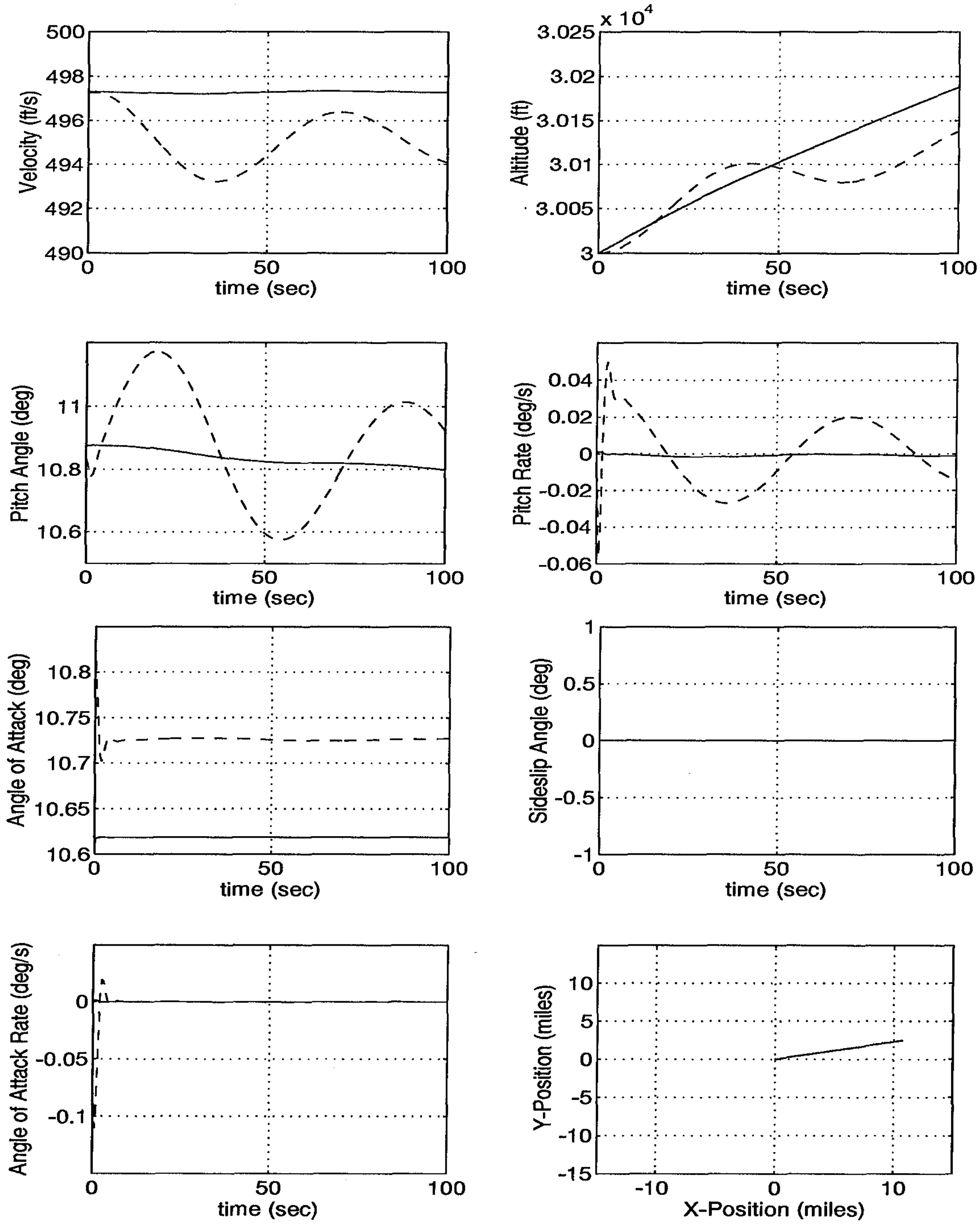


Figure B.3: Flight Point 2—Aircraft Responses to Initial Conditions Set at Trim Values: MATLAB (solid line) and Genesis (dashed line).

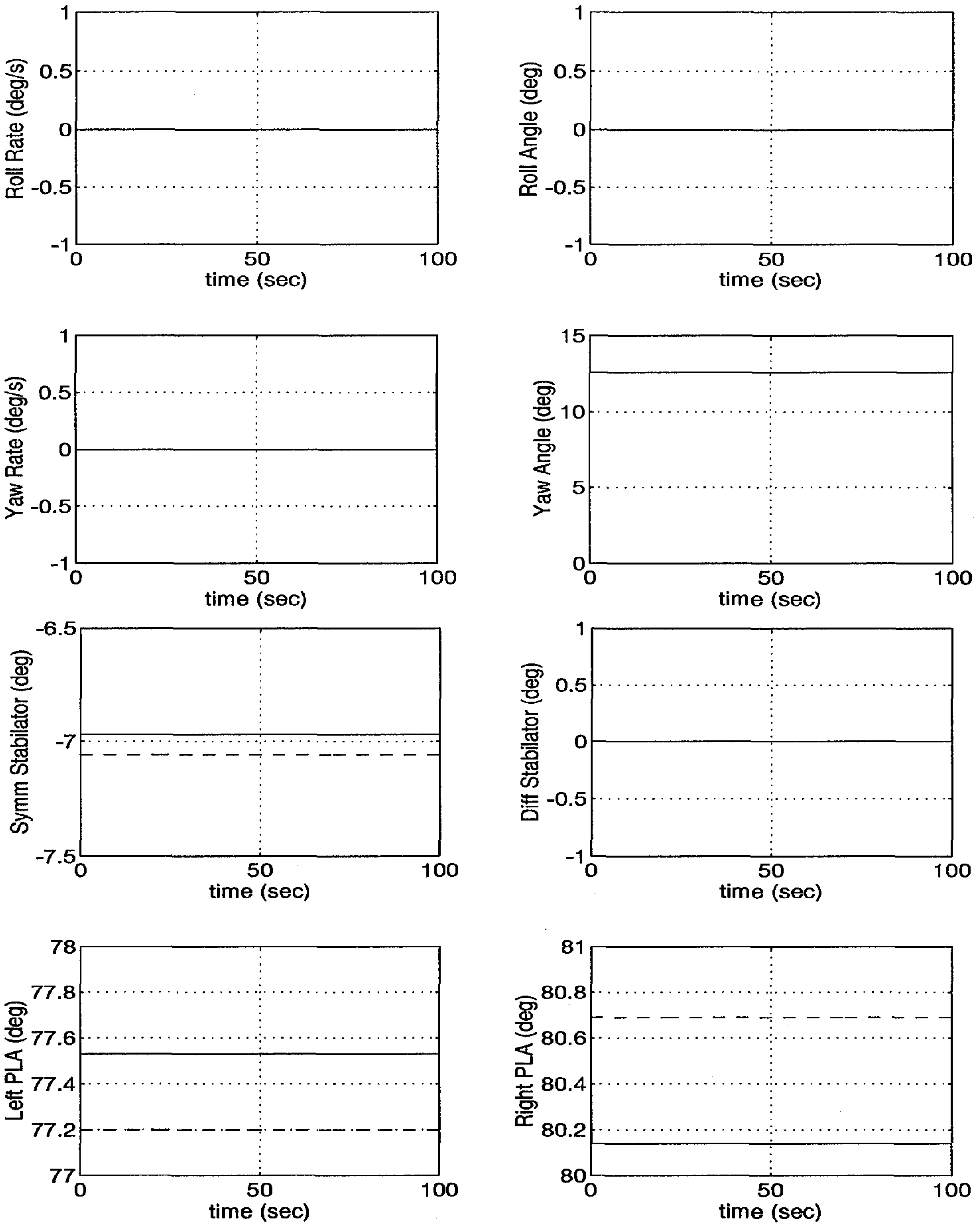


Figure B.4: Flight Point 2—Aircraft Responses to Initial Conditions Set at Trim Values: MATLAB (solid line) and Genesis (dashed line).

Appendix C

LINEARIZED STATE-SPACE MODELS

The following are state-space models of the linearized F-15 model for the two flight conditions listed in Table 1.1.

States = $[\Delta V(ft/sec), \Delta\alpha(rad), \Delta q(rad/sec), \Delta\theta(rad), \Delta p(rad/sec), \Delta\phi(rad),$
$\Delta r(rad/sec), \Delta\psi(rad), \Delta\beta(rad), \Delta h(ft)]^T$

Inputs = $[\Delta\delta_H(deg), \Delta\delta_{PLAL}(deg), \Delta\delta_{PLAR}(deg), \Delta\delta_{Flap}(deg), \Delta\delta_A(deg), \Delta\delta_R(deg)]^T$

Outputs = $\Big[\Delta\dot{V}/g, \Delta V(ft/sec), \Delta h(ft), \Delta\gamma(rad), \Delta q(rad/sec), \Delta\theta(rad), \Delta\alpha(rad),$
$\Delta p(rad/sec), \Delta\phi(rad), \Delta\dot{\beta}(rad/sec), \Delta\beta(rad), \Delta\psi(rad), \Delta\dot{\psi}(rad/sec), \Delta r(rad/sec)\Big]^T$

Flight Point 1

$h = 9,800$ ft, $V_{TAS} = 539$ ft/s

$$A = \left[\begin{array}{rrrrrr}
-0.0137 & -3.461 & 0 & -32.144 & 0 & -1.28^{-5} \\
-2.17^{-4} & -0.779 & 1 & -2.99^{-5} & 0 & -2.97^{-7} \\
4.07^{-4} & -5.802 & -2.501 & -6.50^{-4} & 3.15^{-8} & 4.04^{-8} \\
0 & 0 & 1 & 0 & 0 & 3.23^{-20} \\
7.45^{-22} & 2.42^{-19} & -1.53^{-19} & 0 & -2.21 & 0 \\
0 & 0 & -8.24^{-20} & -3.25^{-20} & 1 & -2.25^{-20} \\
-9.44^{-23} & 9.45^{-20} & 1.88^{-20} & 0 & -7.43^{-2} & 0 \\
0 & 0 & -1.02^{-18} & -2.60^{-21} & 0 & -2.80^{-19} \\
1.16^{-22} & -2.69^{-20} & 0 & 4.40^{-21} & 8.00^{-2} & 0.0594 \\
2.46^{-4} & -539.1 & 0 & 539.1 & 0 & 2.15^{-4}
\end{array}\right. \cdots$$

$$\left.\begin{array}{rrrr}
0 & 0 & -1.04^{-3} & -8.74^{-5} \\
0 & 0 & 1.66^{-12} & 1.38^{-8} \\
-3.15^{-8} & 0 & -4.55^{-13} & -3.78^{-9} \\
1.02^{-18} & 0 & 0 & 0 \\
1.397 & 0 & -27.05 & 0 \\
0.0805 & 0 & 0 & 0 \\
-0.5739 & 0 & 4.675 & 0 \\
1.003 & 0 & 0 & 0 \\
-0.997 & 0 & -0.192 & -1.36^{-27} \\
0 & 0 & -6.63^{-7} & 0
\end{array}\right]$$

$$B = \begin{bmatrix} -.0718 & 0.1067 & 0.1014 & 0 & 0 & 0 \\ -1.57^{-3} & -1.59^{-5} & -1.51^{-5} & 0 & 0 & 0 \\ -0.1508 & 4.34^{-6} & 4.13^{-6} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & -0.169 & 0.0266 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2.24^{-3} & -0.0487 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -1.12^{-24} & 1.67^{-24} & 1.58^{-24} & 0 & -3.83^{-5} & 6.32^{-4} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$C = \left[\begin{array}{rrrrrr} -4.25^{-4} & -0.1077 & 0 & -1 & 0 & -3.99^{-7} \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1.15^{-22} & -2.69^{-20} & 0 & 4.40^{-21} & 0.0800 & 0.0594 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1.03^{-18} & -2.61^{-21} & 0 & -2.80^{-19} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{array} \right. \ldots$$

$$
\left.\begin{array}{rrrr}
0 & 0 & -3.22^{-5} & -2.72^{-6} \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
-0.997 & 0 & -0.192 & -1.36^{-27} \\
0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 \\
1.003 & 0 & 0 & 0 \\
1 & 0 & 0 & 0
\end{array}\right]
$$

$$
D = \left[\begin{array}{rrrrrr}
-2.23^{-3} & 3.31^{-3} & 3.16^{-3} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
-1.12^{-24} & 1.67^{-24} & 1.58^{-24} & 0 & -3.83^{-5} & 6.32^{-4} \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0
\end{array}\right]
$$

Flight Point 2

$h = 30,000$ ft, $V_{TAS} = 497$ ft/s

$$A = \left[\begin{array}{rrrrrr}
-0.0200 & -38.596 & 0 & -32.083 & 0 & -2.90^{-5} \\
-2.61^{-4} & -0.355 & 1 & -2.81^{-4} & 0 & -3.12^{-7} \\
2.65^{-4} & -2.301 & -1.148 & -6.40^{-4} & 3.15^{-8} & 1.13^{-8} \\
0 & 0 & 1 & 0 & 0 & 1.98^{-20} \\
1.37^{-21} & -3.24^{-19} & -9.36^{-20} & 0 & -1.050 & 0 \\
0 & 0 & -1.02^{-20} & -2.05^{-20} & 1 & 2.63^{-23} \\
-1.41^{-22} & 1.74^{-19} & 5.34^{-20} & 0 & -7.52^{-3} & 0 \\
0 & 0 & -5.42^{-20} & -3.86^{-21} & 0 & 1.40^{-22} \\
1.10^{-23} & -7.77^{-20} & 0 & -1.10^{-21} & 0.1842 & 0.0634 \\
4.10^{-3} & -497.3 & 0 & 497.31 & 0 & 4.50^{-4}
\end{array}\right. \cdots$$

$$\left.\begin{array}{rrrr}
0 & 0 & -4.70^{-4} & -2.19^{-4} \\
0 & 0 & 5.37^{-12} & 8.35^{-8} \\
-3.15^{-8} & 0 & -4.94^{-13} & -7.68^{-9} \\
5.33^{-20} & 0 & 0 & 0 \\
1.210 & 0 & -13.163 & 0 \\
0.1917 & 0 & 0 & 0 \\
-0.301 & 0 & 1.455 & 0 \\
1.018 & 0 & 0 & 0 \\
-0.983 & 0 & -0.0944 & -1.20^{-26} \\
0 & 0 & -1.02^{-5} & 0
\end{array}\right]$$

$$B = \begin{bmatrix} -0.142 & 0.0389 & 0.0370 & 0 & 0 & 0 \\ -7.04^{-4} & -1.47^{-5} & -1.39^{-5} & 0 & 0 & 0 \\ -0.0554 & 1.35^{-6} & 1.28^{-6} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.0559 & 5.47^{-3} \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1.40^{-4} & -0.0214 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ -7.77^{-24} & 2.12^{-24} & 2.02^{-24} & 0 & 2.99^{-6} & 3.17^{-4} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$C = \left[\begin{array}{rrrrrr} -6.24^{-4} & -1.203 & 0 & -1 & 0 & -9.05^{-7} \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1.10^{-23} & -7.77^{-20} & 0 & -1.10^{-21} & 0.184 & 0.0633 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -5.42^{-20} & -3.86^{-21} & 0 & 1.40^{-22} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{array} \right. \ldots$$

$$\left.\begin{array}{rrrr}
0 & 0 & -1.46^{-5} & -6.83^{-6} \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
-0.983 & 0 & -0.0944 & -1.20^{-26} \\
0 & 0 & 1 & 0 \\
0 & 1 & 0 & 0 \\
1.018 & 0 & 0 & 0 \\
1 & 0 & 0 & 0
\end{array}\right]$$

$$D = \left[\begin{array}{rrrrrr}
-4.44^{-3} & 1.21^{-3} & 1.15^{-3} & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
-7.77^{-24} & 2.12^{-24} & 2.02^{-24} & 0 & 2.99^{-6} & 3.17^{-4} \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0
\end{array}\right]$$

Appendix D

CLOSED-LOOP MODEL ANALYSIS

D.1 TECS Controller Gains

The following is a summary of the gains obtained for the longitudinal TECS controller by the optimization routine SANDY. The linearized state-space F-15 model for each flight point was used in determining the gains. Refer to Chapter 4 for an explanation of each gain.

Parameter	FlightPoint1	FlightPoint2
K_{EP}	21.57	-93.52
K_{EI}	22.34	72.73
K_{TP}	-102.7	-159.7
K_{TI}	185.0	291.8
K_v	-0.0094	-0.0118
K_h	-1.36^{-4}	-1.07^{-4}
K_θ	94.4008	1000
K_q	34.1758	246.5
K_{GW}	1	1
K_{CAS}	1	1

D.2 Closed-Loop Eigenvalues

A listing of the closed-loop eigenvalues and their corresponding damping and frequencies is given below for both flight points.

Flight Point 1

Eigenvalues	Damping	Frequency(rad/s)
-0.0998	1.000	0.998
$-0.0800 \pm 0.0600i$	0.800	0.100
$-0.0800 \pm 0.0600i$	0.800	0.100
$-0.2926 \pm 0.0962i$	0.950	0.308
-0.3081	1.000	0.308
$-0.3567 \pm 0.3639i$	0.700	0.510
$-0.3672 \pm 0.3748i$	0.700	0.525
-0.7687	1.000	0.769
$-5.0127 \pm 5.1017i$	0.701	7.152
-6.8946	1.000	6.895

Flight Point 2

Eigenvalues	Damping	Frequency(rad/s)
-0.1000	1.000	0.100
$-0.0800 \pm 0.0600i$	0.800	0.100
$-0.0800 \pm 0.0600i$	0.800	0.100
$-0.1235 \pm 0.1260i$	0.700	0.176
$-0.2700 \pm 0.0887i$	0.950	0.284
-0.2842	1.000	0.284
$-0.4308 \pm 0.4395i$	0.700	0.615
-0.7769	1.000	0.777
-4.6791	1.000	4.679
$-8.7309 \pm 7.8011i$	0.746	11.71

D.3 Linearized Model Responses

Figures D.1-D.6 are the command response plots for the linearized F-15 model.

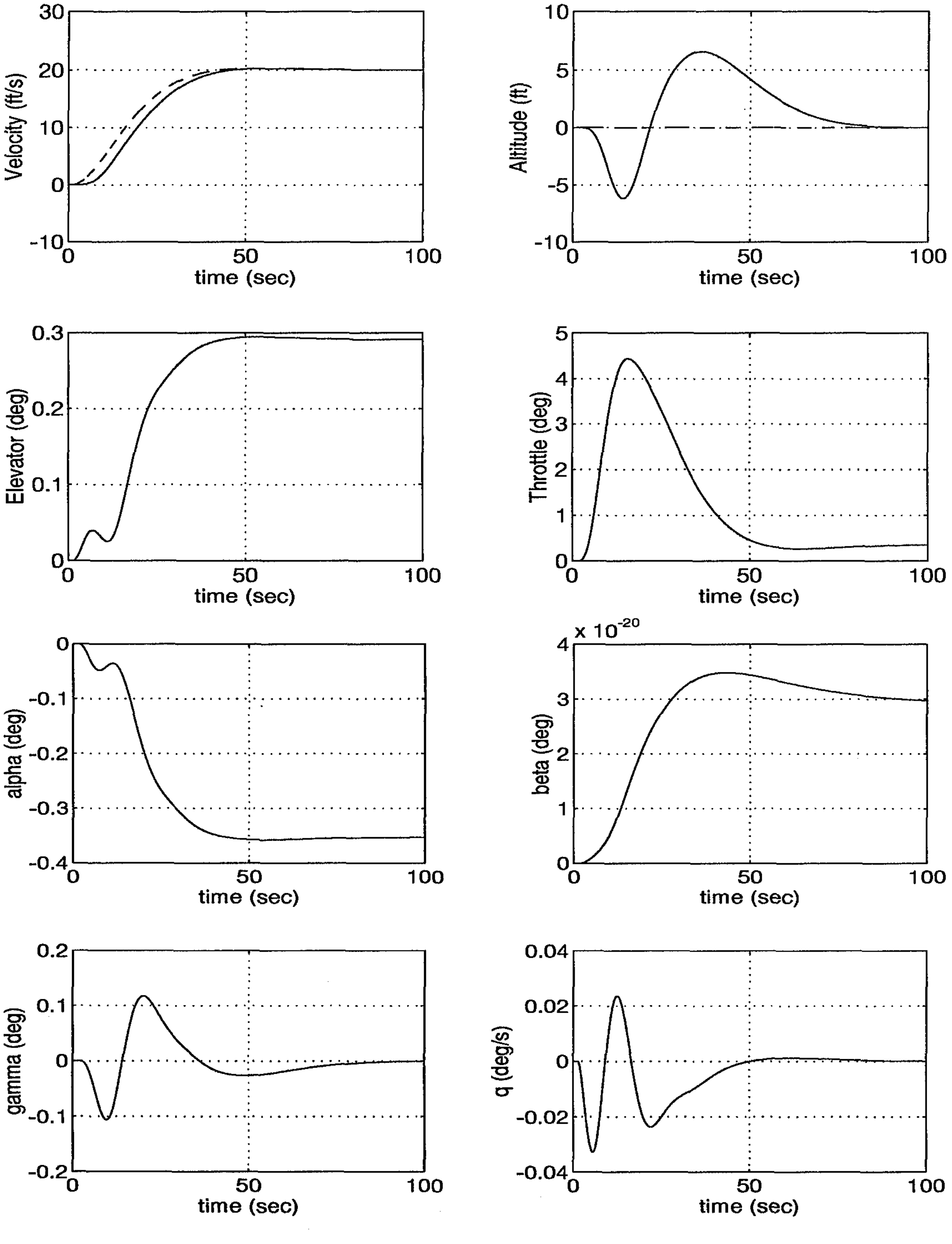


Figure D.1: Flight Point 1—Linear Aircraft Responses to a 20 ft/s Velocity Command.

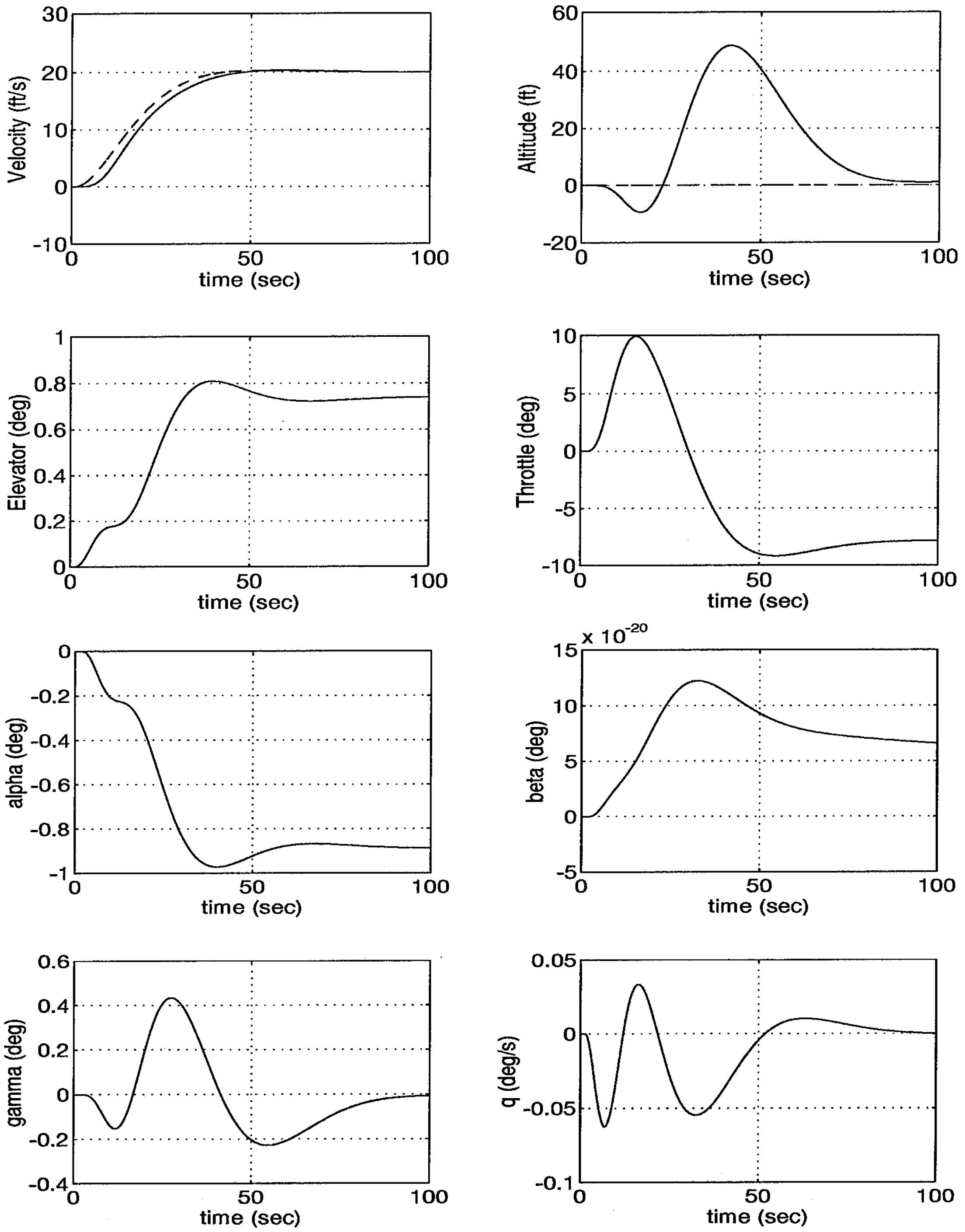


Figure D.2: Flight Point 2—Linear Aircraft Responses to a 20 ft/s Velocity Command.

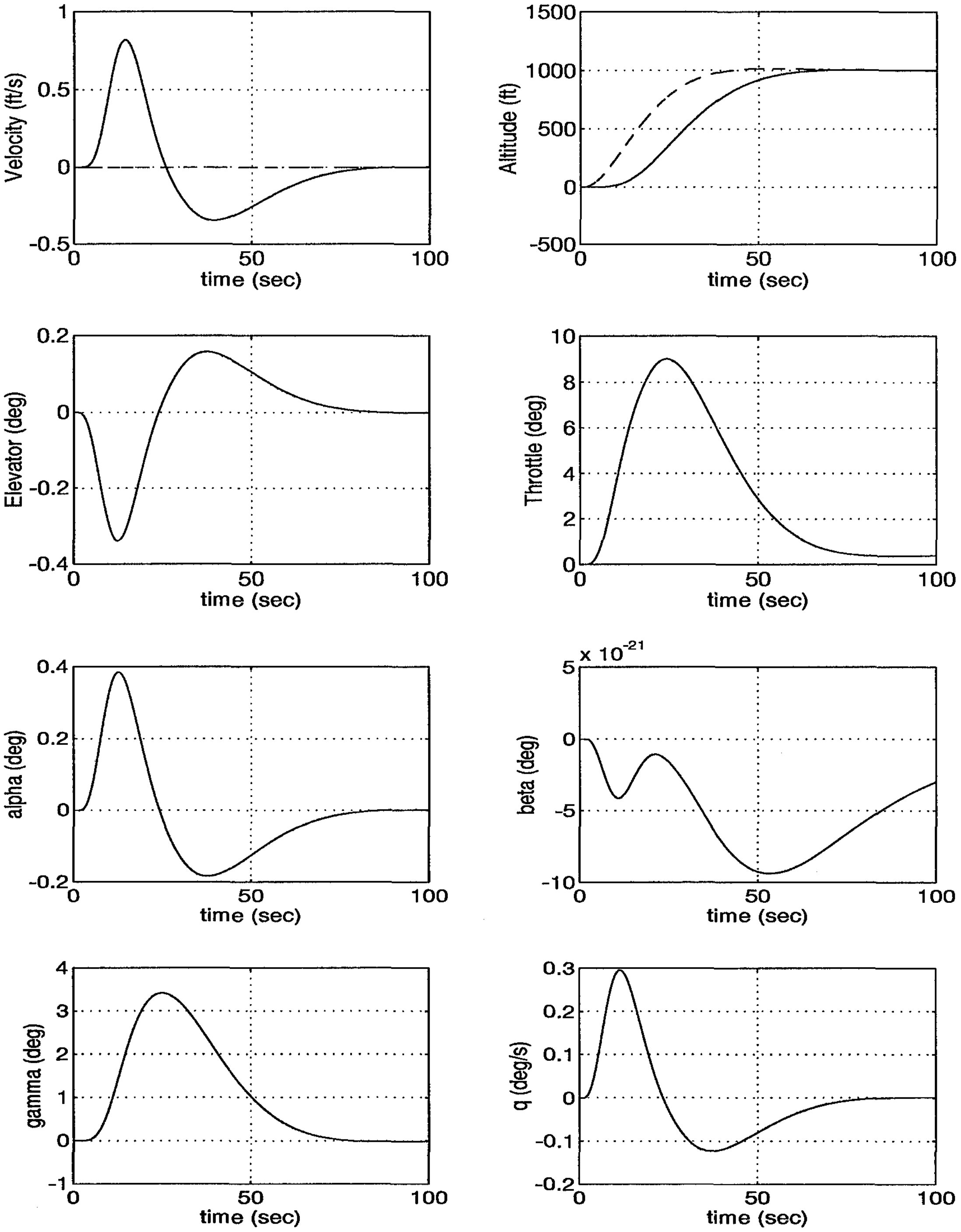


Figure D.3: Flight Point 1—Linear Aircraft Responses to a 1000 ft Altitude Command.

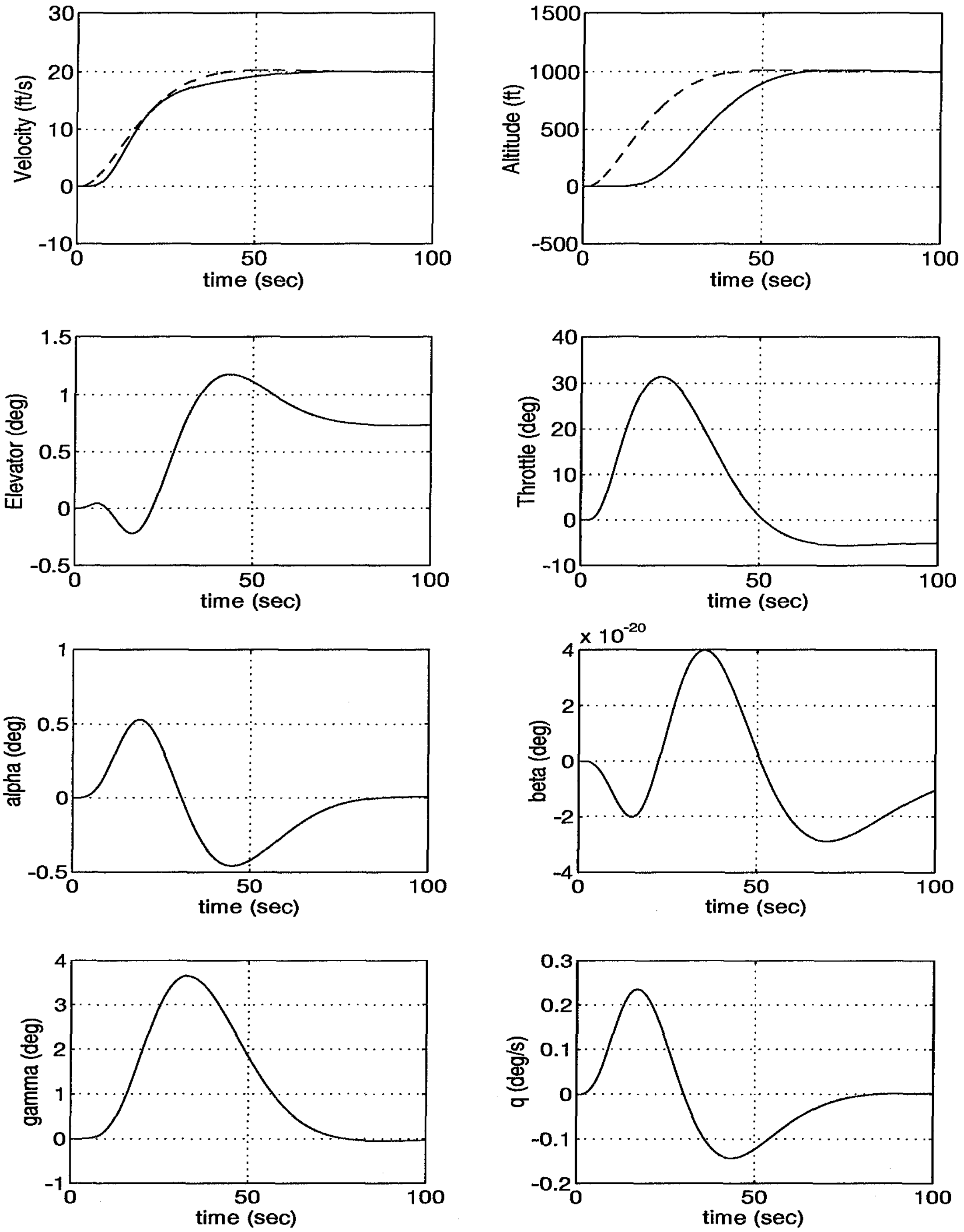


Figure D.4: Flight Point 2—Linear Aircraft Responses to a 1000 ft Altitude Command.

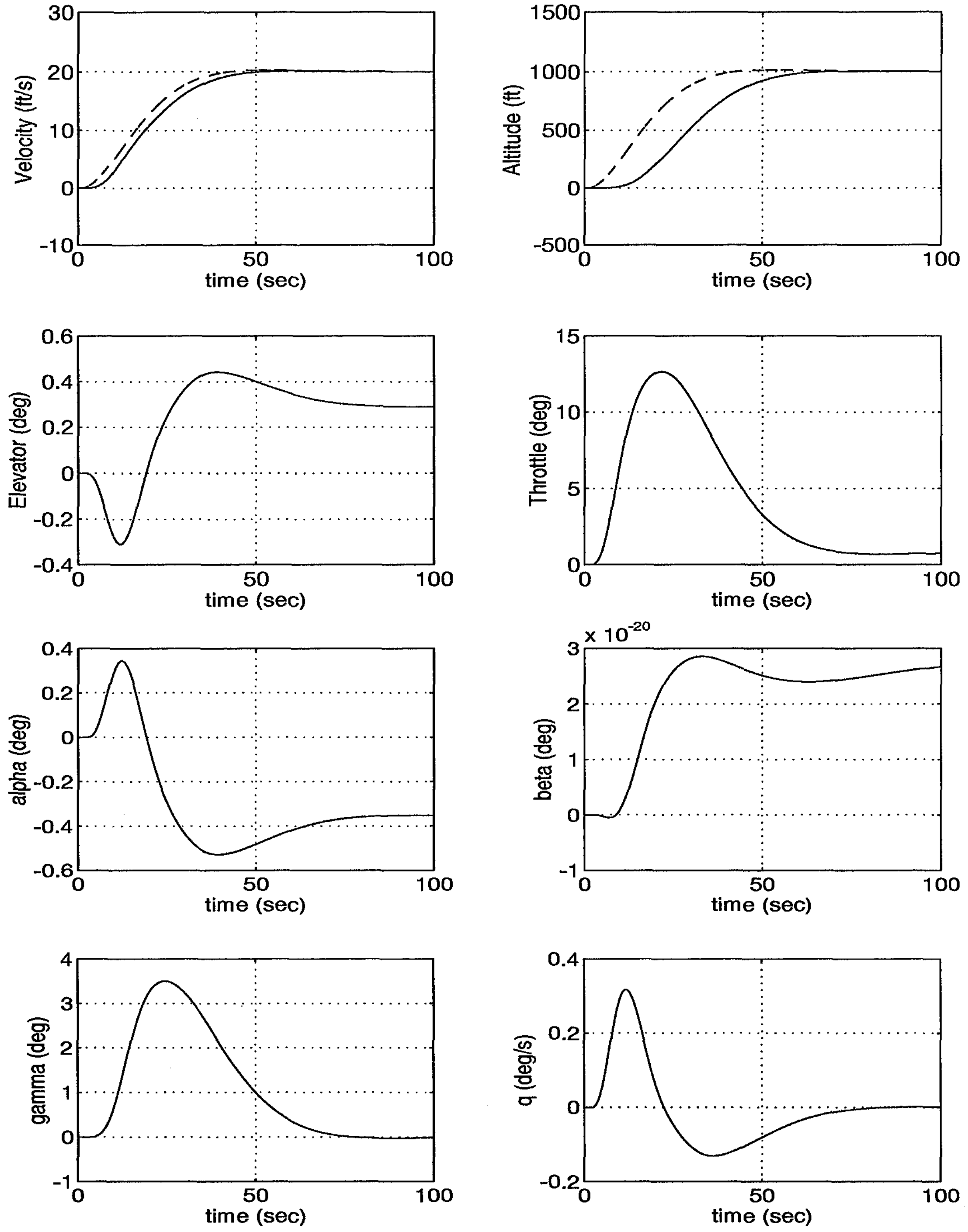


Figure D.5: Flight Point 1—Linear Aircraft Responses to a 20 ft/s Velocity Command and 1000 ft Altitude Command.

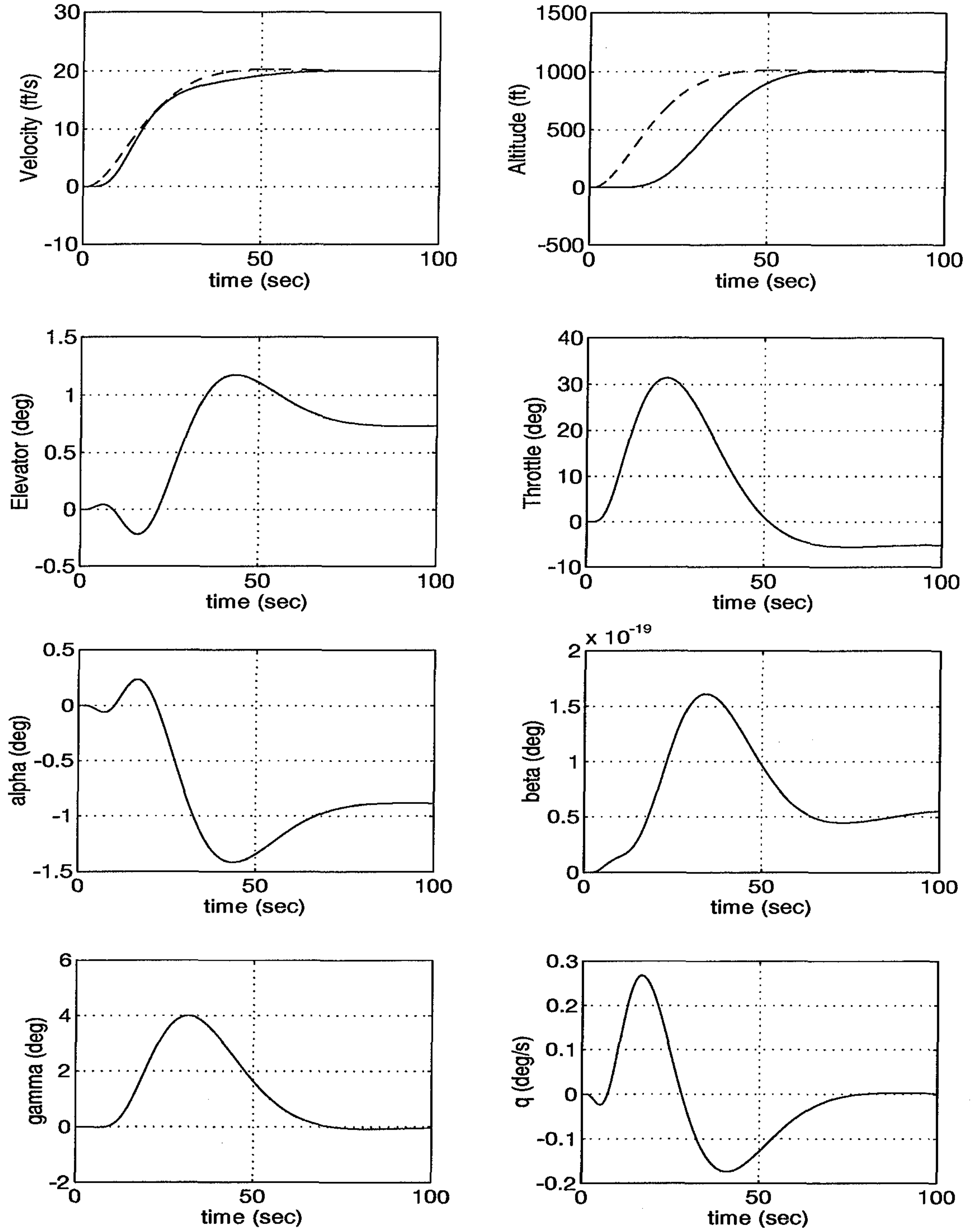


Figure D.6: Flight Point 2—Linear Aircraft Responses to a 20 ft/s Velocity Command and 1000 ft Altitude Command.

D.4 Nonlinear Model Responses

Figures D.7-D.12 are the command response plots for the nonlinear F-15 model.

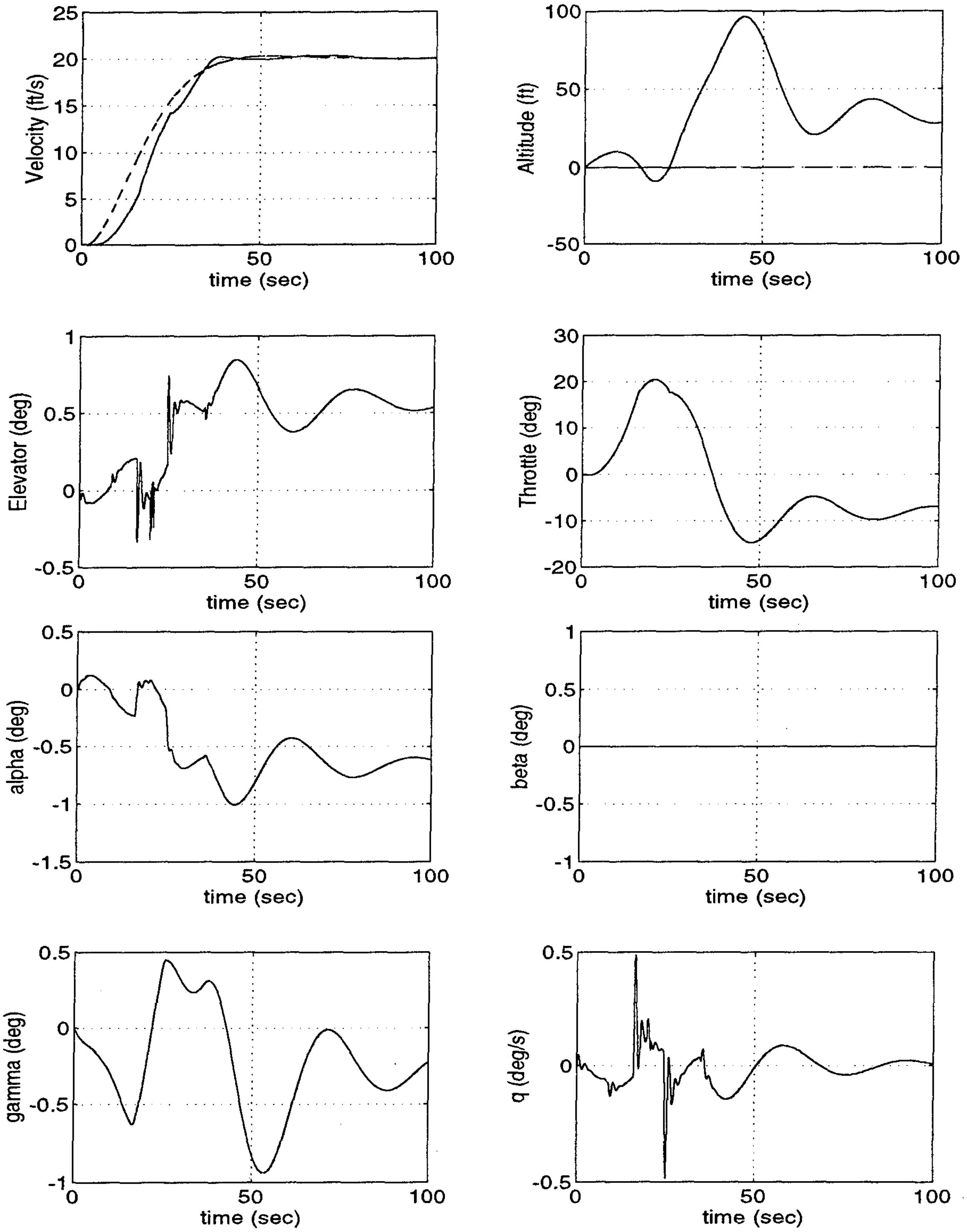


Figure D.7: Flight Point 1—Nonlinear Aircraft Responses to a 20 ft/s Velocity Command.

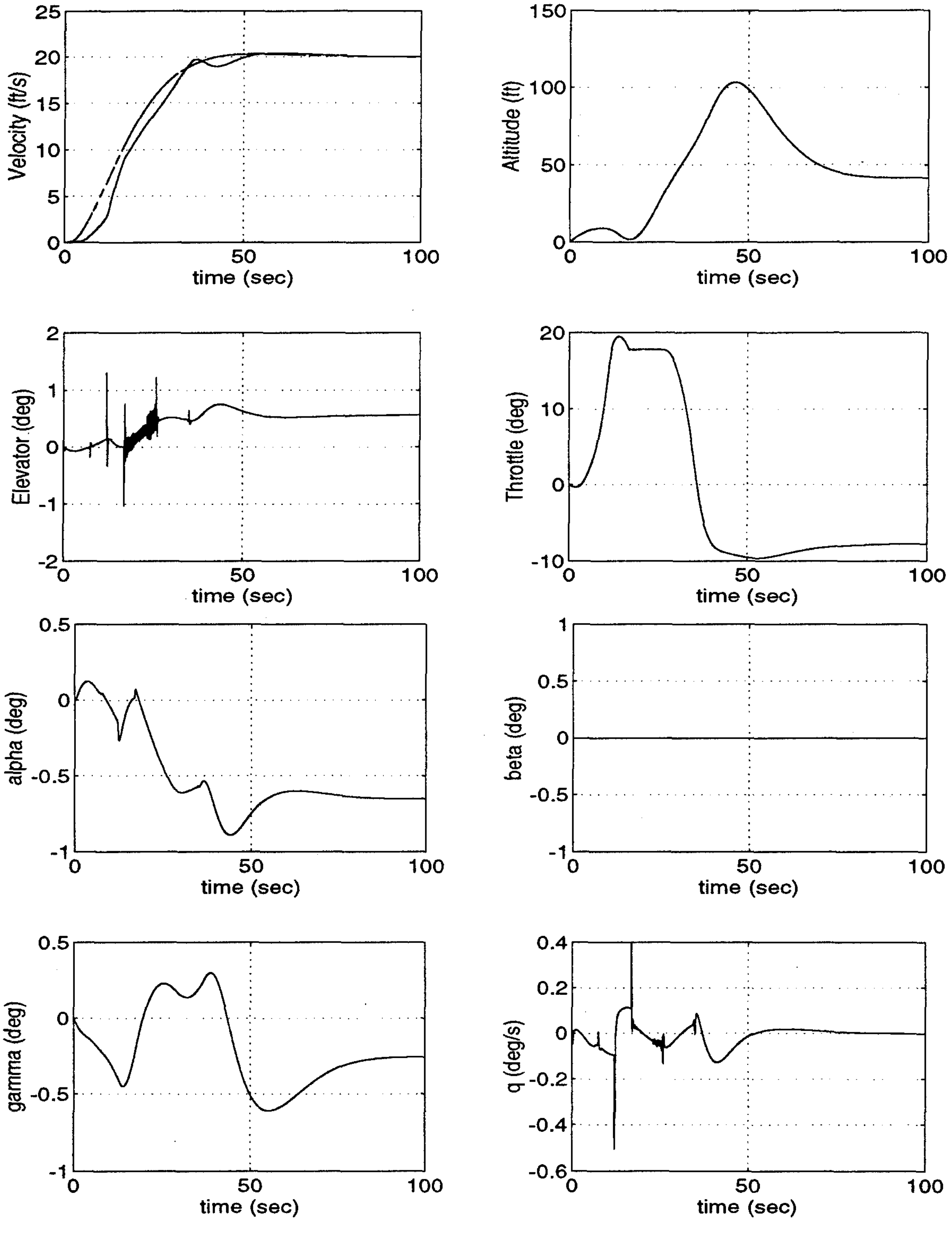


Figure D.8: Flight Point 2—Nonlinear Aircraft Responses to a 20 ft/s Velocity Command.

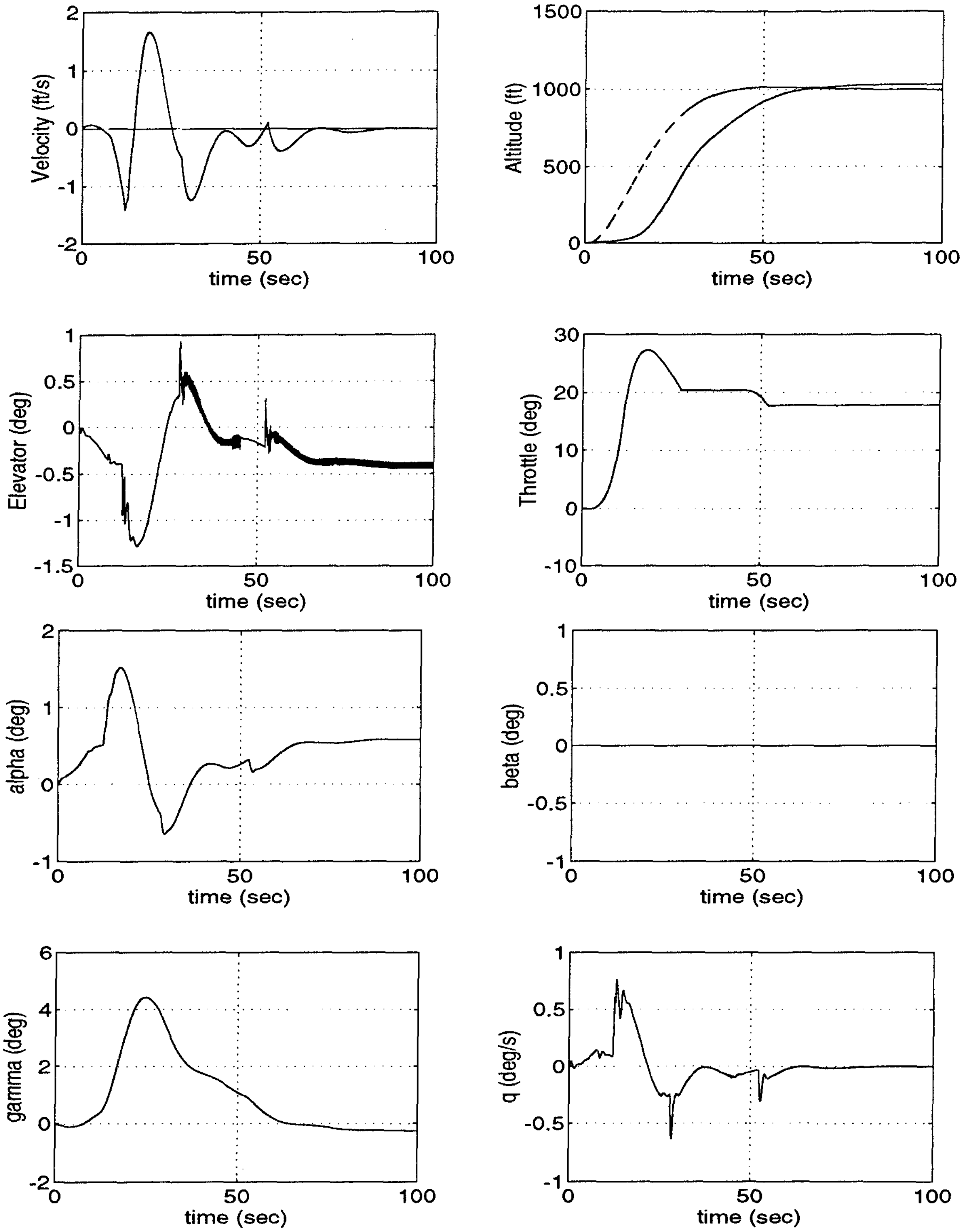


Figure D.9: Flight Point 1—Nonlinear Aircraft Responses to a 1000 ft Altitude Command.

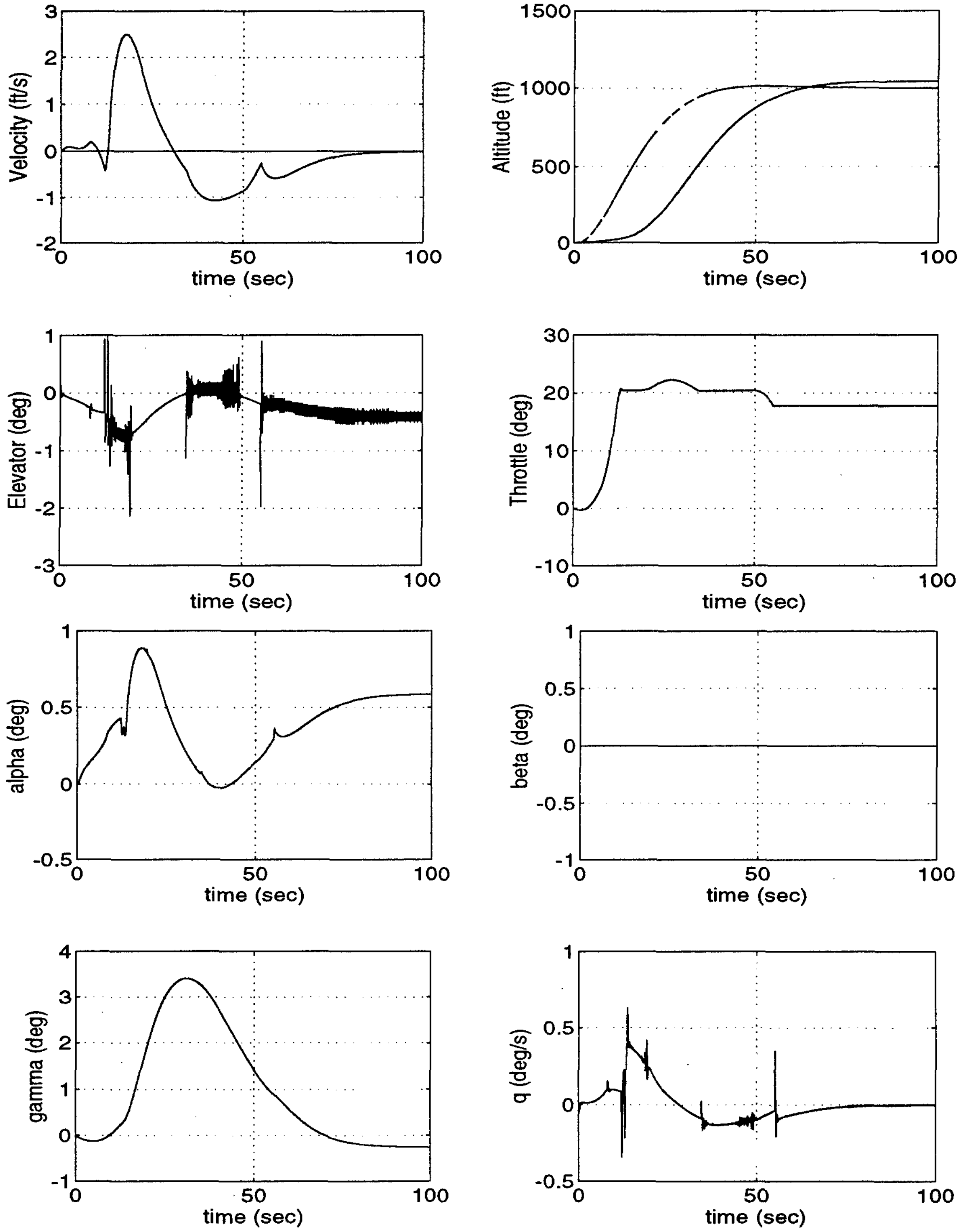


Figure D.10: Flight Point 2—Nonlinear Aircraft Responses to a 1000 ft Altitude Command.

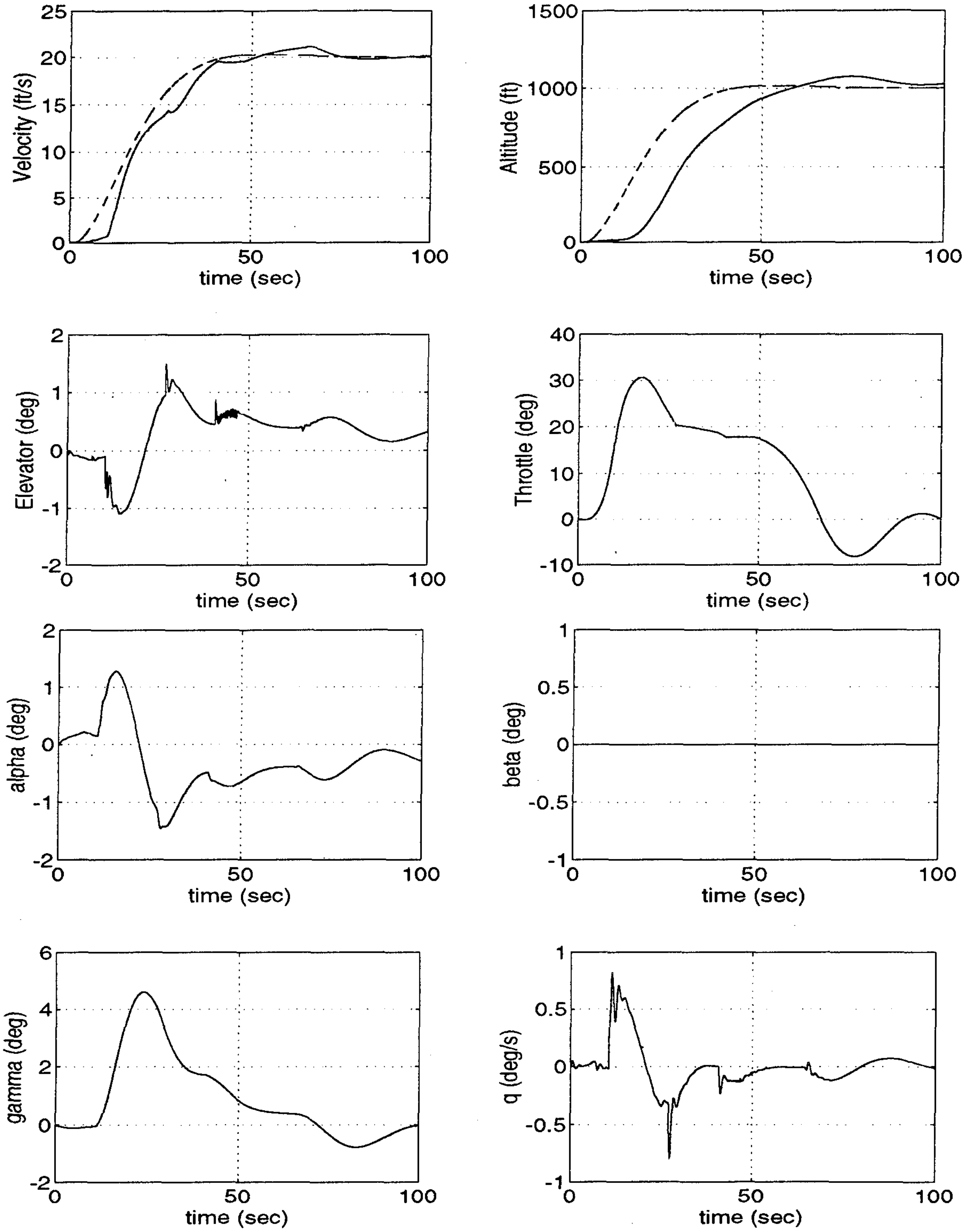


Figure D.11: Flight Point 1—Nonlinear Aircraft Responses to a 20 ft/s Velocity Command and 1000 ft Altitude Command.

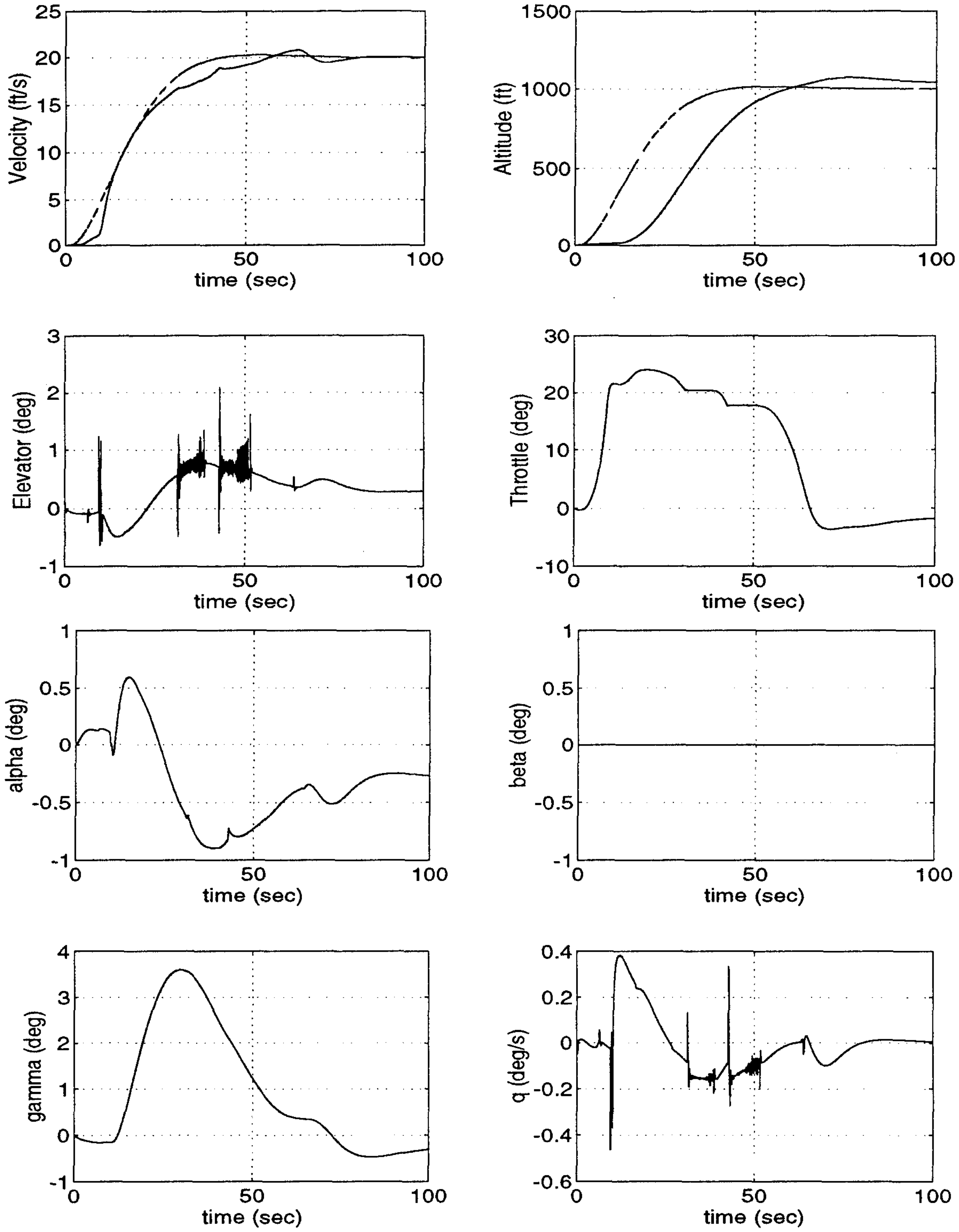


Figure D.12: Flight Point 2—Nonlinear Aircraft Responses to a 20 ft/s Velocity Command and 1000 ft Altitude Command.

Appendix E

F-15 NONLINEAR SIMULATION MODULES

E.1 **A**-*Vector Listing*

The following is a summary of the important **A**-vector parameters used in the F-15 simulation.

```
A(1418) = PLASYM   %%% SYMMETRIC PLA (DEG)           %%%
A(1417) = PLAPR    %%% RIGHT PLA (DEG)               %%%
A(1416) = PLAPL    %%% LEFT  PLA (DEG)               %%%
A(1415) = CN       %%% YAWING MOMENT COEFFICIENT     %%%
A(1414) = CM       %%% PITCHING MOMENT COEFFICIENT   %%%
A(1413) = CL       %%% ROLLING MOMENT COEFFICIENT    %%%
A(1412) = CY       %%% SIDE FORCE COEFFICIENT        %%%
A(1411) = CD       %%% DRAG COEFFICIENT              %%%
A(1410) = CLFT     %%% LIFT COEFFICIENT              %%%
A(1404) = DR       %%% RUDDER DEFLECTION (DEG)       %%%
A(1403) = DD       %%% DIFFERENTIAL STABILATOR (DEG) %%%
A(1402) = DH       %%% SYMMETRIC STABILATOR (DEG)    %%%
A(1401) = DA       %%% AILERON DEFLECTION (DEG)      %%%
A(944)  = psi      %%% HEADING ANGLE (RAD)           %%%
A(943)  = theta    %%% PITCH ANGLE (RAD)             %%%
A(942)  = phi      %%% ROLL ANGLE (RAD)              %%%
A(917)  = betadot  %%% SIDESLIP RATE (RAD/S)         %%%
A(916)  = alpdot   %%% ANGLE OF ATTACK RATE (RAD/S)  %%%
A(915)  = beta     %%% SIDESLIP ANGLE (RAD)          %%%
A(914)  = alpha    %%% ANGLE OF ATTACK (RAD)         %%%
A(863)  = r        %%% YAW RATE (RAD/S)              %%%
A(862)  = q        %%% PITCH RATE (RAD/S)            %%%
A(861)  = p        %%% ROLL RATE (RAD/S)             %%%
```

```
A(830)  = Vdot    %%% ACCELERATION (FT/S^2)            %%%
A(829)  = V       %%% VELOCITY (FT/S)                  %%%
A(825)  = AMCH    %%% MACH NUMBER                      %%%
A(772)  = G       %%% ACCL OF GRAVITY (FT/S^2)         %%%
A(753)  = FPZ     %%% PROPULSION FORCE Z-AXIS (LBS)    %%%
A(752)  = FPY     %%% PROPULSION FORCE Y-AXIS (LBS)    %%%
A(751)  = FPX     %%% PROPULSION FORCE X-AXIS (LBS)    %%%
A(750)  = FAZ     %%% AERO FORCE Z-AXIS (LBS)          %%%
A(749)  = FAY     %%% AERO FORCE Y-AXIS (LBS)          %%%
A(748)  = FAX     %%% AERO FORCE X-AXIS (LBS)          %%%
A(735)  = ANM     %%% YAWING MOMENT (FT-LBS)           %%%
A(734)  = AMM     %%% PITCHING MOMENT (FT-LBS)         %%%
A(733)  = ALM     %%% ROLLING MOMENT (FT-LBS)          %%%
A(716)  = yp      %%% Y-POSITION (FT)                  %%%
A(715)  = xp      %%% X-POSITION (FT)                  %%%
A(713)  = h       %%% ALTITUDE (FT)                    %%%
A(670)  = RHO     %%% DENSITY (SLUGS/FT^3)             %%%
A(669)  = QBAR    %%% DYNAMIC PRESSURE (SL/FT-S^2)     %%%
A(661)  = cbar    %%% WING CHORD (FT)                  %%%
A(660)  = b       %%% WING SPAN (FT)                   %%%
A(659)  = S       %%% WING AREA (FT^2)                 %%%
A(658)  = W       %%% AIRCRAFT WEIGHT (LBS)            %%%
A(639)  = Iyz     %%% PRODUCT OF INERTIA Y-Z PLANE     %%%
A(638)  = Ixy     %%% PRODUCT OF INERTIA X-Y PLANE     %%%
A(637)  = Ixz     %%% PRODUCT OF INERTIA X-Z PLANE     %%%
A(636)  = Iz      %%% MOMENT OF INERTIA Z-AXIS         %%%
A(635)  = Iy      %%% MOMENT OF INERTIA Y-AXIS         %%%
A(634)  = Ix      %%% MOMENT OF INERTIA X-AXIS         %%%
```

E.2 F-15 Nonlinear Aerodynamic Model Listing

The following is a listing of the modules which comprise the nonlinear aerodynamic model for the F-15.

- MATLAB function: `f25aero`

```
function [CLFT,CD,CY,CL,CM,CN,FAX,FAY,FAZ,ALM,AMM,ANM] = ...
          f25aero(A,IA);

%%% DYNAMICS (TRIM, HOLD, RUN, AND LINEARIZATION) %%%
[CLFT,CD,CY,CL,CM,CN,FAX,FAY,FAZ,ALM,AMM,ANM] = ccalc(A);
```

- MATLAB function: `ccalc`

```
%%% ROUTINE TO COMPUTE AERODYNAMIC FORCE AND MOMENT %%%
%%% COEFFICIENTS %%%

%%% WRITTEN JUNE 16,1978
%%% LEE DUKE NASA/DFRC
%%% MODIFIED OCTOBER, 1990  (CLEAN-UP, ADD VAR DECL)
%%% RANDY BRUMBAUGH  PRC
%%% BUG FIXES 12/21/90  RWB
%%% CONVERSION TO MATLAB 12/7/93 JPD

function [CLFT,CD,CY,CL,CM,CN,FAX,FAY,FAZ,ALM,AMM,ANM] = ccalc(A)

%%% ASSIGN A-ARRAY VAR NAMES (INDEX + 1001) %%%
DGR    = A(981);  GO     = A(994);  ALP    = A(914);
BTA    = A(915);  B      = A(660);  V      = A(829);
CBAR   = A(661);  H      = A(713);  VDOT   = A(830);
ALPDOT = A(916);  BTADOT = A(917);  Q      = A(862);
P      = A(861);  R      = A(863);  G      = A(772);
AZ     = A(765);  DA     = A(1401); DH     = A(1402);
```

```
DT     = A(1403); DR     = A(1404); DSB    = A(1405);
PHI    = A(942);  THA    = A(943);  QBAR   = A(669);
S      = A(659);  DELX   = A(654);  DELY   = A(656);
DELZ   = A(655);  PLAPL  = A(1416); PLAPR  = A(1417);
AMCH   = A(825);
DSB45 = 0.0;

%%% PLA IS AVERAGE OF ENGINE THROTTLE SETTINGS %%%
PLA = (PLAPL+PLAPR)/2;
ALPD = ALP*DGR;
BTAD = BTA*DGR;
if V ~= 0
B2V  = B/(2.0*V);
CB2V = CBAR/(2.0*V);
end
COSALP = cos(ALP);
SINALP = sin(ALP);
COSTHA = cos(THA);
COSPHI = cos(PHI);
ALP1 = ALPD;
if ALP1 <  0,  ALP1 =  0.0; end
if ALP1 > 10,  ALP1 = 10.0; end
ALP2 = ALPD;
if ALPD < 10,  ALP2 = 10.0; end

LATEST = 0;  %%% LOGICAL 'FALSE' %%%
if ALPD >= 25
  if abs(BTAD) <= 10
    if AMCH <= 0.6,  LATEST = 1; end   %%% LOGICAL 'TRUE' %%%
  end;
end;

DCDALT = 0.0005/10000.0*(30000.0-H);
```

```
CDNOZK = 1.0;
if PLA > 83,  CDNOZK = 0.0; end

%%% DO TABLE LOOK-UP %%%
[FMCOEF] = tlu(A);
CLFTB  = FMCOEF(1);  DCLNZ = FMCOEF(2);  DCLO  = FMCOEF(3);
DCLA1  = FMCOEF(4);  DCLA2 = FMCOEF(5);  CDB   = FMCOEF(6);
DCDNOZ = FMCOEF(7);  DCDSB = FMCOEF(8);  CYB1  = FMCOEF(9);
CYB2   = FMCOEF(10); CYDA  = FMCOEF(11); CYDD  = FMCOEF(12);
DCYDR  = FMCOEF(13); DRYK  = FMCOEF(14); CLB1  = FMCOEF(15);
CLB2   = FMCOEF(16); CLP   = FMCOEF(17); CLR   = FMCOEF(18);
CLDA   = FMCOEF(19); CLDD  = FMCOEF(20); DCLDR = FMCOEF(21);
DRLK   = FMCOEF(22); DCLSB = FMCOEF(23); CMB   = FMCOEF(24);
DCMNZ  = FMCOEF(25); DCMO  = FMCOEF(26); DNOSB = FMCOEF(27);
DNO    = FMCOEF(28); CMQ   = FMCOEF(29); CMAD  = FMCOEF(30);
CNB1   = FMCOEF(31); CNB2  = FMCOEF(32); CNP   = FMCOEF(33);
CNR    = FMCOEF(34); CNDA  = FMCOEF(35); CNDD  = FMCOEF(36);
CNDR   = FMCOEF(37); DRNK  = FMCOEF(38); DCNSB = FMCOEF(39);

%%% DETERMINE TERMS TO BE USED IN FORCE AND MOMENT EQUATIONS %%%
CYB = CYB1;
if LATEST == 1,  CYB = CYB2; end
CLB = CLB1;
if LATEST == 1
  if PLA < 35,  CLB = CLB2; end
end;
CNB = CNB1;
if LATEST == 1,  CNB = CNB2; end
ANZC = (AZ-G*COSTHA*COSPHI)/GO;

%%% COMPUTE AERODYNAMIC FORCE AND MOMENT COEFFICIENTS %%%
   %%% RWB ADDED 0.95 %%%
   CLFT = 0.95*CLFTB+DCLNZ*ANZC+DSB45*(DCLO+DCLA1*ALP1+DCLA2*...
          (ALP2-10.0));
   CY    = CYB+CYDA*DA+CYDD*DT-DCYDR*DRYK;
   CL    = CLB+CLDA*DA+CLDD*DT-DCLDR*DRLK+B2V*(CLP*P+CLR*R)+...
          DCLSB*DSB45;
   CDX = CDB;
   if ALPD > 32,  CDX = CLFTB*SINALP/COSALP; end
   if ALPD < 40
     if ALPD > 32,  CDX = (CDX-CDB)/8.0*(ALPD-32.0)+CDB; end
   end;
   %%% RWB ADDED 1.02 %%%
   CD = 1.02*CDX+DCDALT+DCDNOZ*CDNOZK+DSB45*DCDSB;

   %%% CALCULATES ALPHADOT FOR PITCHING MOMENT %%%
   CM = CMB+DCMNZ*ANZC+CB2V*(CMQ*Q+CMAD*ALPDOT)+CLFTB*DNO+...
        DSB45*(DCMO+DNOSB*CLFTB);
   CN = CNB+CNDA*DA+CNDD*DT+CNDR*DR*DRNK+B2V*(CNP*P+CNR*R)+...
        DCNSB*DSB45;

%%% ADDED TO CORRECT FORCE AND MOMENT COEFFICIENTS FROM  %%%
%%% REFERENCE POINT TO THE A/C CG.  THESE LINES OF CODE  %%%
%%% WERE MERGED INTO THIS SUBROUTINE FROM THE SUBROUTINE %%%
%%% 'CGCALC' WHICH HAS BEEN DELETED.                     %%%

%%% COMPUTE BODY AXIS FORCE COEFFICIENTS %%%
    FX = -CD*COSALP+CLFT*SINALP;
    FY = CY;
    FZ = -CD*SINALP-CLFT*COSALP;

%%% COMPUTE MOMENTS INDUCED BY CHANGE OF REFERENCE %%%
    DELTL = (FZ*DELY-FY*DELZ)/B;
    DELTM = (FX*DELZ-FZ*DELX)/CBAR;
    DELTN = (FY*DELX-FX*DELY)/B;
```

```
%%% CORRECT MOMENT COEFFICIENTS %%%
    CL = CL+DELTL;
    CM = CM+DELTM;
    CN = CN+DELTN;

%%% CALCULATE FORCE TERMS %%%
    FAX = FX*QBAR*S;
    FAY = FY*QBAR*S;
    FAZ = FZ*QBAR*S;

%%% CALCULATE MOMENT TERMS %%%
    ALM = QBAR*S*B*CL;
    AMM = QBAR*S*CBAR*CM;
    ANM = QBAR*S*B*CN;
```

- MATLAB function: `tlu`

```
function [FMCOEF] = tlu(A)

%%% ROUTINE TO DO TABLE LOOK-UP FOR AEROMODEL %%%

%%% WRITTEN JUNE 1, 1978  %%%
%%% LEE DUKE NASA/DFRC    %%%
%%% MODIFIED OCTOBER 1990 %%%
%%% RANDY BRUMBAUGH  PRC  %%%

%%% DECLARE GLOBAL VARIABLES %%%
global F101A F105A F108A F107A F106A F201A F203A F206A F301A
global F302A F313A F314A F315A F316A F401A F402A F411A F412A
global F413A F414A F415A F416A F418A F501A F505A F506A F507A
global F509A F513A F514A F601A F602A F611A F612A F613A F614A
global F615A F617A F618A

%%% ASSIGN A-ARRAY VAR NAMES %%%
```

```
DR   = A(1404);
ALP  = A(914);
BTA  = A(915);
DGR  = A(981);
AMCH = A(825);
DH   = A(1402);

%%% CREATE DATA ARRAYS %%%
AMCHA  = [0.2 0.4 0.6 0.8 0.9 1.0 1.1 1.2 1.4 1.6];
AMCHB  = [0.2 0.4 0.6 0.8 1.0 1.2 1.4 1.6];
ALPHAA = [-12.0 -8.0 -4.0 0.0 4.0 8.0 12.0 16.0];
   ALPHAA = [ALPHAA, 20.0 24.0 28.0 32.0 36.0 40.0 ...
             44.0 48.0 52.0 56.0 60.0];
ALPHAB = [25.0 30.0 35.0 40.0 45.0 50.0 55.0 60.0];
DHX    = [-25.0 -15.0 -5.0 5.0 15.0];
CLX    = [-1.0 -0.8 -0.6 -0.4 -0.2 0.0 0.2 0.4 0.6 ...
          0.8 1.0 1.2 1.4 1.6];
BETAA  = [0.0 4.0 8.0 12.0 16.0 20.0 24.0 28.0];
BETAB  = [-10.0 -5.0 0.0 5.0 10.0];
BETAC  = [0.0 4.0 8.0 12.0 16.0];
BETAD  = [10.0 20.0 30.0];
DRX1   = [0.0 10.0 20.0 30.0];
ADR    = abs(DR);

%%% CONVERT TO DEGREES %%%
ALPD   = ALP*DGR;
BTAD   = BTA*DGR;
ABSBTA = abs(BTAD);

%%% CALCULATE INDICES INTO ARRAYS %%%
IDXMA1 = 1;
for I = 2:9,
   if AMCH >= AMCHA(I), IDXMA1 = I; end
```

```
end;
IDXMA2 = IDXMA1+1;
XDXMB1 = AMCH/0.2+0.01;
IDXMB1 = floor(XDXMB1);
if IDXMB1 < 1,  IDXMB1 = 1; end
if IDXMB1 > 7,  IDXMB1 = 7; end
IDXMC1 = 1;
XDXAA1 = (ALPD+12.0)/4.0+1.01;
IDXAA1 = floor(XDXAA1);
if IDXAA1 < 1, IDXAA1 = 1; end
if IDXAA1 > 18, IDXAA1 = 18; end
XDXAB1 = (ALPD-25.0)/5.0+1.01;
IDXAB1 = floor(XDXAB1);
if IDXAB1 < 1, IDXAB1 = 1; end
if IDXAB1 > 7, IDXAB1 = 7; end
XDXDH1 = (DH+25.0)/10.0+1.01;
IDXDH1 = floor(XDXDH1);
if IDXDH1 < 1, IDXDH1 = 1; end
if IDXDH1 > 4, IDXDH1 = 4; end
XDXBA1 = ABSBTA/4.0+1.01;
IDXBA1 = floor(XDXBA1);
if IDXBA1 < 1, IDXBA1 = 1; end
if IDXBA1 > 7, IDXBA1 = 7; end
XDXBB1 = (BTAD+10.0)/5.0+1.01;
IDXBB1 = floor(XDXBB1);
if IDXBB1 < 1, IDXBB1 = 1; end
if IDXBB1 > 4, IDXBB1 = 4; end
IDXBC1 = IDXBA1;
if IDXBC1 < 1, IDXBC1 = 1; end
if IDXBC1 > 4, IDXBC1 = 4; end
IDXBD1 = 1;
if ABSBTA > 10.0, IDXBD1 = 2; end
XDXDR1 = ADR/10.0+1.01;
```

```
IDXDR1 = floor(XDXDR1);
if IDXDR1 < 1, IDXDR1 = 1; end
if IDXDR1 > 3, IDXDR1 = 3; end
INDXA1 = IDXMA1+10*(IDXAA1-1+19*(IDXDH1-1));
INDXA2 = INDXA1+1;
INDXA3 = INDXA1+10;
INDXA4 = INDXA2+10;
INDXA5 = INDXA1+190;
INDXA6 = INDXA2+190;
INDXA7 = INDXA3+190;
INDXA8 = INDXA4+190;
INDXC1 = IDXMA1+10*(IDXAA1-1);
INDXC2 = INDXC1+1;
INDXC3 = INDXC1+10;
INDXC4 = INDXC2+10;
INDXD1 = IDXMA1+10*(IDXAA1-1+19*(IDXBA1-1));
INDXD2 = INDXD1+1;
INDXD3 = INDXD1+10;
INDXD4 = INDXD2+10;
INDXD5 = INDXD1+190;
INDXD6 = INDXD2+190;
INDXD7 = INDXD3+190;
INDXD8 = INDXD4+190;
INDXE1 = IDXMC1+2*(IDXAB1-1+8*(IDXBB1-1));
INDXE2 = INDXE1+1;
INDXE3 = INDXE1+2;
INDXE4 = INDXE2+2;
INDXE5 = INDXE1+16;
INDXE6 = INDXE2+16;
INDXE7 = INDXE3+16;
INDXE8 = INDXE4+16;
INDXF1 = IDXMA1+10*(IDXAA1-1+19*(IDXDR1-1));
INDXF2 = INDXF1+1;
```

```
INDXF3 = INDXF1+10;
INDXF4 = INDXF2+10;
INDXF5 = INDXF1+190;
INDXF6 = INDXF2+190;
INDXF7 = INDXF3+190;
INDXF8 = INDXF4+190;
INDXG1 = IDXMA1+10*(IDXAA1-1+19*(IDXBC1-1));
INDXG2 = INDXG1+1;
INDXG3 = INDXG1+10;
INDXG4 = INDXG2+10;
INDXG5 = INDXG1+190;
INDXG6 = INDXG2+190;
INDXG7 = INDXG3+190;
INDXG8 = INDXG4+190;
INDXH1 = IDXMA1+10*(IDXAA1-1+19*(IDXBD1-1));
INDXH2 = INDXH1+1;
INDXH3 = INDXH1+10;
INDXH4 = INDXH2+10;
INDXH5 = INDXH1+190;
INDXH6 = INDXH2+190;
INDXH7 = INDXH3+190;
INDXH8 = INDXH4+190;

%%% COMPUTE INTERPOLATION RATIOS %%%
RATOMA = (AMCH-AMCHA(IDXMA1))/(AMCHA(IDXMA2)-AMCHA(IDXMA1));
RATOMB = (AMCH-AMCHB(IDXMB1))/0.2;
RATOMC = (AMCH-0.2)/0.4;
RATOAA = (ALPD-ALPHAA(IDXAA1))/4.0;
RATOAB = (ALPD-ALPHAB(IDXAB1))/5.0;
RATODH = (DH*0.8-DHX(IDXDH1))/10.0;
RATOBA = (ABSBTA-BETAA(IDXBA1))/4.0;
RATOBB = (BTAD-BETAB(IDXBB1))/5.0;
RATOBC = (ABSBTA-BETAC(IDXBC1))/4.0;
```

```
RATOBD = (ABSBTA-BETAD(IDXBD1))/10.0;
if ABSBTA < 10.0, RATOBD = 0.0; end
RATODR = (ADR-DRX1(IDXDR1))/10.0;
if AMCH   > 1.6,   RATOMA = 1.0; end
if AMCH   > 1.6,   RATOMB = 1.0; end
if AMCH   < 0.2,   RATOMC = 0.0; end
if AMCH   > 0.6,   RATOMC = 1.0; end
if ALPD   < -12.0, RATOAA = 0.0; end
if ALPD   > 60.0,  RATOAA = 1.0; end
if ALPD   < 25.0,  RATOAB = 0.0; end
if ALPD   > 60.0,  RATOAB = 1.0; end
if ABSBTA > 28.0,  RATOBA = 1.0; end
if BTAD   < -10.0, RATOBB = 0.0; end
if BTAD   > 10.0,  RATOBB = 1.0; end
if ABSBTA > 16.0,  RATOBC = 1.0; end
if ABSBTA > 30.0,  RATOBD = 1.0; end
RATMA1 = 1.0-RATOMA;
RATMC1 = 1.0-RATOMC;
RATAA1 = 1.0-RATOAA;
RATAB1 = 1.0-RATOAB;
RATDH1 = 1.0-RATODH;
RATBA1 = 1.0-RATOBA;
RATBB1 = 1.0-RATOBB;
RATBC1 = 1.0-RATOBC;
RATBD1 = 1.0-RATOBD;
RATDR1 = 1.0-RATODR;

%%% TABLE LOOK UP %%%
F101A1 = RATOMA*F101A(INDXA2)+F101A(INDXA1)*RATMA1;
F101A2 = RATOMA*F101A(INDXA4)+F101A(INDXA3)*RATMA1;
F101A3 = RATOMA*F101A(INDXA6)+F101A(INDXA5)*RATMA1;
F101A4 = RATOMA*F101A(INDXA8)+F101A(INDXA7)*RATMA1;
F101B1 = RATOAA*F101A2+F101A1*RATAA1;
```

```
F101B2 = RATOAA*F101A4+F101A3*RATAA1;
F101B1 = RATOAA*F101A2+F101A1*RATAA1;
CLFTB  = RATODH*F101B2+F101B1*RATDH1;
DCLNZ  = RATOMA*F105A(IDXMA2)+F105A(IDXMA1)*RATMA1;
DCL0   = RATOMA*F106A(IDXMA2)+F106A(IDXMA1)*RATMA1;
DCLA1  = RATOMA*F107A(IDXMA2)+F107A(IDXMA1)*RATMA1;
DCLA2  = RATOMA*F108A(IDXMA2)+F108A(IDXMA1)*RATMA1;
DCDNOZ = RATOMA*F203A(IDXMA2)+F203A(IDXMA1)*RATMA1;
DCDSB  = RATOMA*F206A(IDXMA2)+F206A(IDXMA1)*RATMA1;
F301A1 = RATOMA*F301A(INDXD2)+F301A(INDXD1)*RATMA1;
F301A2 = RATOMA*F301A(INDXD4)+F301A(INDXD3)*RATMA1;
F301A3 = RATOMA*F301A(INDXD6)+F301A(INDXD5)*RATMA1;
F301A4 = RATOMA*F301A(INDXD8)+F301A(INDXD7)*RATMA1;
F301B1 = RATOAA*F301A2+F301A1*RATAA1;
F301B2 = RATOAA*F301A4+F301A3*RATAA1;
F301   = RATOBA*F301B2+F301B1*RATBA1;
CYB1   = F301*sign(BTA);
F302A1 = RATOMC*F302A(INDXE2)+F302A(INDXE1)*RATMC1;
F302A2 = RATOMC*F302A(INDXE4)+F302A(INDXE3)*RATMC1;
F302A3 = RATOMC*F302A(INDXE6)+F302A(INDXE5)*RATMC1;
F302A4 = RATOMC*F302A(INDXE8)+F302A(INDXE7)*RATMC1;
F302B1 = RATOAB*F302A2+F302A1*RATAB1;
F302B2 = RATOAB*F302A4+F302A3*RATAB1;
CYB2   = RATOBB*F302B2+F302B1*RATBB1;
F313A1 = RATOMA*F313A(INDXC2)+F313A(INDXC1)*RATMA1;
F313A2 = RATOMA*F313A(INDXC4)+F313A(INDXC3)*RATMA1;
CYDA   = RATOAA*F313A2+F313A1*RATAA1;
F314A1 = RATOMA*F314A(INDXC2)+F314A(INDXC1)*RATMA1;
F314A2 = RATOMA*F314A(INDXC4)+F314A(INDXC3)*RATMA1;
CYDD   = RATOAA*F314A2+F314A1*RATAA1;
F315A1 = RATOMA*F315A(INDXF2)+F315A(INDXF1)*RATMA1;
F315A2 = RATOMA*F315A(INDXF4)+F315A(INDXF3)*RATMA1;
F315A3 = RATOMA*F315A(INDXF6)+F315A(INDXF5)*RATMA1;
```

```
F315A4 = RAT0MA*F315A(INDXF8)+F315A(INDXF7)*RATMA1;
F315B1 = RAT0AA*F315A2+F315A1*RATAA1;
F315B2 = RAT0AA*F315A4+F315A3*RATAA1;
F315   = RAT0DR*F315B2+F315B1*RATDR1;
DCYDR  = F315*sign(DR);
DRYK   = RAT0MA*F316A(IDXMA2)+F316A(IDXMA1)*RATMA1;
F401A1 = RAT0MA*F401A(INDXD2)+F401A (INDXD1)*RATMA1;
F401A2 = RAT0MA*F401A(INDXD4)+F401A (INDXD3)*RATMA1;
F401A3 = RAT0MA*F401A(INDXD6)+F401A (INDXD5)*RATMA1;
F401A4 = RAT0MA*F401A(INDXD8)+F401A (INDXD7)*RATMA1;
F401B1 = RAT0AA*F401A2+F401A1*RATAA1;
F401B2 = RAT0AA*F401A4+F401A3*RATAA1;
F401   = RAT0BA*F401B2+F401B1*RATBA1;
CLB1   = F401*sign(BTA);
F402A1 = RAT0MC*F402A(INDXE2)+F402A (INDXE1)*RATMC1;
F402A2 = RAT0MC*F402A(INDXE4)+F402A (INDXE3)*RATMC1;
F402A3 = RAT0MC*F402A(INDXE6)+F402A (INDXE5)*RATMC1;
F402A4 = RAT0MC*F402A(INDXE8)+F402A (INDXE7)*RATMC1;
F402B1 = RAT0AB*F402A2+F402A1*RATAB1;
F402B2 = RAT0AB*F402A4+F402A3*RATAB1;
CLB2   = RAT0BB*F402B2+F402B1*RATBB1;
F411A1 = RAT0MA*F411A(INDXC2)+F411A(INDXC1)*RATMA1;
F411A2 = RAT0MA*F411A(INDXC4)+F411A(INDXC3)*RATMA1;
CLP    = RAT0AA*F411A2+F411A1*RATAA1;
F412A1 = RAT0MA*F412A(INDXC2)+F412A(INDXC1)*RATMA1;
F412A2 = RAT0MA*F412A(INDXC4)+F412A(INDXC3)*RATMA1;
CLR    = RAT0AA*F412A2+F412A1*RATAA1;
F413A1 = RAT0MA*F413A(INDXC2)+F413A(INDXC1)*RATMA1;
F413A2 = RAT0MA*F413A(INDXC4)+F413A(INDXC3)*RATMA1;
CLDA   = RAT0AA*F413A2+F413A1*RATAA1;
F414A1 = RAT0MA*F414A(INDXC2)+F414A(INDXC1)*RATMA1;
F414A2 = RAT0MA*F414A(INDXC4)+F414A(INDXC3)*RATMA1;
CLDD   = RAT0AA*F414A2+F414A1*RATAA1;
```

```
F415A1 = RATOMA*F415A(INDXF2)+F415A(INDXF1)*RATMA1;
F415A2 = RATOMA*F415A(INDXF4)+F415A(INDXF3)*RATMA1;
F415A3 = RATOMA*F415A(INDXF6)+F415A(INDXF5)*RATMA1;
F415A4 = RATOMA*F415A(INDXF8)+F415A(INDXF7)*RATMA1;
F415B1 = RATOAA*F415A2+F415A1*RATAA1;
F415B2 = RATOAA*F415A4+F415A3*RATAA1;
F415   = RATODR*F415B2+F415B1*RATDR1;
DCLDR  = F415*sign(DR);
DRLK   = RATOMA*F416A(IDXMA2)+F416A(IDXMA1)*RATMA1;
F418A1 = RATOMA*F418A(INDXG2)+F418A(INDXG1)*RATMA1;
F418A2 = RATOMA*F418A(INDXG4)+F418A(INDXG3)*RATMA1;
F418A3 = RATOMA*F418A(INDXG6)+F418A(INDXG5)*RATMA1;
F418A4 = RATOMA*F418A(INDXG8)+F418A(INDXG7)*RATMA1;
F418B1 = RATOAA*F418A2+F418A1*RATAA1;
F418B2 = RATOAA*F418A4+F418A3*RATAA1;
DCLSB  = RATOBC*F418B2+F418B1*RATBC1;
F501A1 = RATOMA*F501A(INDXA2)+F501A(INDXA1)*RATMA1;
F501A2 = RATOMA*F501A(INDXA4)+F501A(INDXA3)*RATMA1;
F501A3 = RATOMA*F501A(INDXA6)+F501A(INDXA5)*RATMA1;
F501A4 = RATOMA*F501A(INDXA8)+F501A(INDXA7)*RATMA1;
F501B1 = RATOAA*F501A2+F501A1*RATAA1;
F501B2 = RATOAA*F501A4+F501A3*RATAA1;
F501B1 = RATOAA*F501A2+F501A1*RATAA1;
CMB    = RATODH*F501B2+F501B1*RATDH1;
DCMNZ  = RATOMA*F505A(IDXMA2)+F505A(IDXMA1)*RATMA1;
DCMO   = RATOMA*F506A(IDXMA2)+F506A(IDXMA1)*RATMA1;
DNOSB  = RATOMA*F507A(IDXMA2)+F507A(IDXMA1)*RATMA1;
DNO    = RATOMA*F509A(IDXMA2)+F509A(IDXMA1)*RATMA1;
F513A1 = RATOMA*F513A(INDXC2)+F513A(INDXC1)*RATMA1;
F513A2 = RATOMA*F513A(INDXC4)+F513A(INDXC3)*RATMA1;
CMQ    = RATOAA*F513A2+F513A1*RATAA1;
F514A1 = RATOMA*F514A(INDXC2)+F514A(INDXC1)*RATMA1;
F514A2 = RATOMA*F514A(INDXC4)+F514A(INDXC3)*RATMA1;
```

```
CMAD   = RATOAA*F514A2+F514A1*RATAA1;
F601A1 = RATOMA*F601A(INDXD2)+F601A(INDXD1)*RATMA1;
F601A2 = RATOMA*F601A(INDXD4)+F601A(INDXD3)*RATMA1;
F601A3 = RATOMA*F601A(INDXD6)+F601A(INDXD5)*RATMA1;
F601A4 = RATOMA*F601A(INDXD8)+F601A(INDXD7)*RATMA1;
F601B1 = RATOAA*F601A2+F601A1*RATAA1;
F601B2 = RATOAA*F601A4+F601A3*RATAA1;
F601   = RATOBA*F601B2+F601B1*RATBA1;
CNB1   = F601*sign(BTA);
F602A1 = RATOMC*F602A(INDXE2)+F602A(INDXE1)*RATMC1;
F602A2 = RATOMC*F602A(INDXE4)+F602A(INDXE3)*RATMC1;
F602A3 = RATOMC*F602A(INDXE6)+F602A(INDXE5)*RATMC1;
F602A4 = RATOMC*F602A(INDXE8)+F602A(INDXE7)*RATMC1;
F602B1 = RATOAB*F602A2+F602A1*RATAB1;
F602B2 = RATOAB*F602A4+F602A3*RATAB1;
CNB2   = RATOBB*F602B2+F602B1*RATBB1;
F611A1 = RATOMA*F611A(INDXC2)+F611A(INDXC1)*RATMA1;
F611A2 = RATOMA*F611A(INDXC4)+F611A(INDXC3)*RATMA1;
CNP    = RATOAA*F611A2+F611A1*RATAA1;
F612A1 = RATOMA*F612A(INDXC2)+F612A(INDXC1)*RATMA1;
F612A2 = RATOMA*F612A(INDXC4)+F612A(INDXC3)*RATMA1;
CNR    = RATOAA*F612A2+F612A1*RATAA1;
F613A1 = RATOMA*F613A(INDXC2)+F613A(INDXC1)*RATMA1;
F613A2 = RATOMA*F613A(INDXC4)+F613A(INDXC3)*RATMA1;
CNDA   = RATOAA*F613A2+F613A1*RATAA1;
F614A1 = RATOMA*F614A(INDXC2)+F614A(INDXC1)*RATMA1;
F614A2 = RATOMA*F614A(INDXC4)+F614A(INDXC3)*RATMA1;
CNDD   = RATOAA*F614A2+F614A1*RATAA1;
F615A1 = RATOMA*F615A(INDXH2)+F615A(INDXH1)*RATMA1;
F615A2 = RATOMA*F615A(INDXH4)+F615A(INDXH3)*RATMA1;
F615A3 = RATOMA*F615A(INDXH6)+F615A(INDXH5)*RATMA1;
F615A4 = RATOMA*F615A(INDXH8)+F615A(INDXH7)*RATMA1;
F615B1 = RATOMA*F615A2+F615A1*RATMA1;
```

```
F615B2 = RATOMA*F615A4+F615A3*RATMA1;
CNDR   = RATOBD*F615B2+F615B1*RATBD1;
F617A1 = RATOMA*F617A(INDXC2)+F617A(INDXC1)*RATMA1;
F617A2 = RATOMA*F617A(INDXC4)+F617A(INDXC3)*RATMA1;
DRNK   = RATOAA*F617A2+F617A1*RATAA1;
F618A1 = RATOMA*F618A(INDXG2)+F618A(INDXG1)*RATMA1;
F618A2 = RATOMA*F618A(INDXG4)+F618A(INDXG3)*RATMA1;
F618A3 = RATOMA*F618A(INDXG6)+F618A(INDXG5)*RATMA1;
F618A4 = RATOMA*F618A(INDXG8)+F618A(INDXG7)*RATMA1;
F618B1 = RATOAA*F618A2+F618A1*RATAA1;
F618B2 = RATOAA*F618A4+F618A3*RATAA1;
DCNSB  = RATOBC*F618B2+F618B1*RATBC1;

%%% LOOK UP CD AS A FUNCTION OF CLFT BASIC (F101) %%%
XDXCL1 = (CLFTB+1.0)/0.2+1.01;
IDXCL1 = floor(XDXCL1);
if IDXCL1 < 1,  IDXCL1 = 1;  end
if IDXCL1 > 13, IDXCL1 = 13; end
INDXB1 = IDXMA1+10*(IDXCL1-1);
INDXB2 = INDXB1+1;
INDXB3 = INDXB1+10;
INDXB4 = INDXB2+10;
RATOCL = (CLFTB-CLX(IDXCL1))/0.2;
RATCL1 = 1.0-RATOCL;
F201A1 = RATOMA*F201A(INDXB2)+F201A(INDXB1)*RATMA1;
F201A2 = RATOMA*F201A(INDXB4)+F201A(INDXB3)*RATMA1;
CDB    = (RATOCL*F201A2+F201A1*RATCL1);

%%% CREATE 'FMCOEF' VECTOR %%%
FMCOEF(1)  = CLFTB;  FMCOEF(2)  = DCLNZ;  FMCOEF(3)  = DCL0;
FMCOEF(4)  = DCLA1;  FMCOEF(5)  = DCLA2;  FMCOEF(6)  = CDB;
FMCOEF(7)  = DCDNOZ; FMCOEF(8)  = DCDSB;  FMCOEF(9)  = CYB1;
FMCOEF(10) = CYB2;   FMCOEF(11) = CYDA;   FMCOEF(12) = CYDD;
FMCOEF(13) = DCYDR;  FMCOEF(14) = DRYK;   FMCOEF(15) = CLB1;
FMCOEF(16) = CLB2;   FMCOEF(17) = CLP;    FMCOEF(18) = CLR;
FMCOEF(19) = CLDA;   FMCOEF(20) = CLDD;   FMCOEF(21) = DCLDR;
FMCOEF(22) = DRLK;   FMCOEF(23) = DCLSB;  FMCOEF(24) = CMB;
FMCOEF(25) = DCMNZ;  FMCOEF(26) = DCMO;   FMCOEF(27) = DNOSB;
FMCOEF(28) = DNO;    FMCOEF(29) = CMQ;    FMCOEF(30) = CMAD;
FMCOEF(31) = CNB1;   FMCOEF(32) = CNB2;   FMCOEF(33) = CNP;
FMCOEF(34) = CNR;    FMCOEF(35) = CNDA;   FMCOEF(36) = CNDD;
FMCOEF(37) = CNDR;   FMCOEF(38) = DRNK;   FMCOEF(39) = DCNSB;
```

E.3 F-15 Nonlinear Propulsion Model Listing

The following is a listing of the modules which comprise the nonlinear propulsion model for the F-15.

- MATLAB function: `f25eng`

```
function [FPX,FPY,FPZ,DCL,DCM,DCN,TAUL,TAUR,PLAL,PLAR,COUT1C,...
          COUT2C,FIRST] = f25eng(A,IA);

%%% INITIALIZATION SECTION %%%
   STEP = A(1000);
   engdin(STEP);

%%% DYNAMICS (TRIM, HOLD, RUN, AND LINEARIZATION) %%%
   [FPX,FPY,FPZ,DCL,DCM,DCN,TAUL,TAUR,PLAL,PLAR,COUT1C,...
    COUT2C,FIRST] = engine(A,IA);
```

- MATLAB function: `engdin`

```
function engdin(STEP);

%%% ROUTINE TO READ IN ENGINE DATA FOR ENGINE MODEL %%%
%%% WRITTEN AUG. 1, 1978    %%%
%%%     LEE DUKE NASA DFRC  %%%

%%% COMMON BLOCKS
      %%% ENGVAR
global C03RTN C03RTP C05F1K C06LMN C06LMP B02LMN B02LMP
global B03RTN B03RTP B04F1K B06F1E B06F1K
      %%% ENGTAB
global C03DAT C03PA C03NA C07DAT B03DAT B03PA B03NA B07DAT...
       FIDLAX FMILAX FMAXAX
global TIDLA TMILA TMAXA FIDLA FMILA FMAXA

      HI = STEP;
      G060 = 115826.4;
      arg = 0;
[C03DAT,C07DAT,B03DAT,B07DAT,TIDLA,FIDLAX,TMILA,FMILAX,TMAXA,...
 FMAXAX] = engdat(arg);
FIDLAX = FIDLAX/G060;
FMILAX = FMILAX/G060;
FMAXAX = FMAXAX/G060;

%%% DEFINE CONSTANTS %%%
      C05F1C =  1.0;
      C06LMN =  0.0;
      C06LMP =  1.0;
      B02LMN = -0.02;
      B02LMP =  0.02;
      B04F1C =  1.0;
      B06F1C = 20.0;

%%% ENTRY ENGDN1 %%%
%%% CALCULATIONS DEPENDENT ON FRAME TIME %%%
      C05F1K = (C05F1C*HI)/2.0;
      B04F1K = (B04F1C*HI)/2.0;
      B06F1E = exp(-B06F1C*HI);
      B06F1K = (1.0-B06F1E)/2.0;
      FIDLA = FIDLAX;
      FMILA = FMILAX;
      FMAXA = FMAXAX;

%%% ADDED, DUTTON 1/20/94
      C03PA = C03DAT(1:4);
      C03NA = C03DAT(5:8);
      B03PA = B03DAT(1:4);
      B03NA = B03DAT(5:8);
```

- MATLAB function: engine

```
function [FPX,FPY,FPZ,DCL,DCM,DCN,TAUL,TAUR,PLAL,PLAR,...
          COUT1C,COUT2C,FIRST] = engine(A,IA);

%%% ROUTINE TO COMPUTE EFFECTS OF ENGINE(S)
%%% WRITTEN FEBRUARY 1,1983
%%%     LEE DUKE NASA/DFRF
%%% MODIFIED FOR DIFFERENT ANGLE DEFINITION JUNE, 1985
%%%     JOE PAHLE  NASA/DFRF
%%% MODIFIED 9/91
%%%     JIM BUFFINGTON WL/FIGC
%%% VARIABLES DEFINITIONS:
%%% THRUST(I,J)  THE TOTAL THRUST OF THE I-TH
%%% ENGINE (LBS)
%%% TLOCAT(I,J)  THE DISPLACEMENT OF THE I-TH ENGINE
%%%                     FROM THE AIRCRAFT REFERENCE
%%%         C.G. (FEET), WHERE
%%% J=1 CORRESPONDS TO THE
%%%     THE X-BODY AXIS
%%%     COORDINATE OF THE
%%%     I-TH ENGINE
%%% J=2 CORRESPONDS TO THE
%%%     THE Y-BODY AXIS
%%%     COORDINATE OF THE
%%%     I-TH ENGINE
%%% J=3 CORRESPONDS TO THE
%%%     THE Z-BODY AXIS
%%%     COORDINATE OF THE
%%%     I-TH ENGINE
%%%  XYANGL(I)   THE ANGLE IN THE X-Y BODY AXIS
%%%        PLANE FROM THE X-BODY AXIS TO
%%%                    THE PROJECTION OF THE I-TH ENGINE
%%%                    AXIS ONTO THE X-Y PLANE. (DEG)
%%% XZANGL(I)   THE ANGLE BETWEEN THE X-Y BODY AXIS
%%%                    PLANE AND THE I-TH ENGINE AXIS
%%%                    PROJECTED ON THE X-Z' BODY/ENGINE AXIS
%%% EIX(I)      THE MOMENT OF INERTIA ABOUT THE
%%%                X-ENGINE AXIS OF THE I-TH ENGINE
%%%                    (SLUG-FT**2).
%%% AMSENG(I)   THE MASS OF THE ROTATING MACHINERY
%%%                    IN THE I-TH ENGINE (SLUGS)
%%% ENGOMG(I)   THE ROTATIONAL VELOCITY OF THE
%%%                    I-TH ENGINE. POSITIVE ROTATION IS
%%%                    MEASURED USING THE R.H.R. ABOUT THE
%%%                    X-ENGINE AXIS. (RAD/SEC)
%%% TVANXY(I)   THE ANGLE IN THE X-Y ENGINE AXIS
%%%           PLANE FROM THE X-ENGINE AXIS TO
%%%                    THE PROJECTION OF THE I-TH THRUST VECTOR
%%%                    ONTO THE X-Y ENGINE PLANE. (DEG)
%%% TVANXZ(I)   THE ANGLE BETWEEN THE X-Y ENGINE AXIS
%%%                    PLANE AND THE I-TH THRUST VECTOR
%%%                    PROJECTED ON THE X-Z' ENGINE/THRUST AXIS
%%%     DXTHRS(I)   THE DISTANCE BETWEEN THE C.G. OF THE
%%%                    ENGINE AND THE THRUST POINT MEASURED
%%%                    POSITIVE IN THE NEGATIVE ENGINE X-AXIS
%%%                             DIRECTION
%%% FPX     TOTAL THRUST IN THE X-BODY AXIS
%%% FPY     TOTAL THRUST IN THE Y-BODY AXIS
%%% FPZ     TOTAL THRUST IN THE Z-BODY AXIS
%%%     DCM         TOTAL PITCHING MOMENT INCREMENT
%%%                    DUE TO ENGINES
%%%     DCL         TOTAL ROLLING MOMENT INCREMENT
%%%                    DUE TO ENGINES
%%%     DCN         TOTAL YAWING MOMENT INCREMENT
%%%                    DUE TO ENGINES

%%% COMMON BLOCK
global THRUST TLOCAT XYANGL XZANGL TVANXY TVANXZ DXTHRS EIX
global AMSENG ENGOMG

%%% ASSIGN A-ARRAY VAR NAMES %%%
DGR = A(981);
Q   = A(862);
P   = A(861);
R   = A(863);

%%% CALL USER ENGINE MODEL INTERFACE ROUTINE %%%
% FIRST was originally set to Logical 1 stored in data IA(502)
[TAUL,TAUR,PLAL,PLAR,COUT1C,COUT2C,FIRST] = uengin(A,IA);

A(1419) = TAUL;    A(1420) = TAUR;   A(1431) = PLAL;
A(667)  = COUT1C;  A(668)  = COUT2C; IA(502)=FIRST;
A(1432) = PLAR;

%%% COMPUTE COMPONENTS OF THRUST %%%
   for I = 1:4,
         ANGLXZ = XZANGL(I)/DGR;
         ANGLXY = XYANGL(I)/DGR;
         CSANXZ(I) = cos(ANGLXZ);
         SNANXZ(I) = sin(ANGLXZ);
         CSANXY(I) = cos(ANGLXY);
         SNANXY(I) = sin(ANGLXY);
         TANGXY = TVANXY(I)/DGR;
         TANGXZ = TVANXZ(I)/DGR;
         CSTAXY = cos(TANGXY);
         SNTAXY = sin(TANGXY);
         CSTAXZ = cos(TANGXZ);
         SNTAXZ = sin(TANGXZ);
         XTHRSI =  THRUST(I)*CSTAXZ*CSTAXY;
         YTHRSI =  THRUST(I)*CSTAXZ*SNTAXY;
         ZTHRSI = -THRUST(I)*SNTAXZ;
         XTHRST(I) = XTHRSI*CSANXZ(I)*CSANXY(I)-YTHRSI*...
                     SNANXY(I)+ZTHRSI*SNANXZ(I)*CSANXY(I);
         YTHRST(I) = XTHRSI*CSANXZ(I)*SNANXY(I)+YTHRSI*...
                     CSANXY(I)+ZTHRSI*SNANXZ(I)*CSANXY(I);
         ZTHRST(I) = -XTHRSI*SNANXZ(I)+ZTHRSI*CSANXZ(I);
    end

%%% COMPUTE TOTAL X-AXIS AND Z-AXIS THRUST %%%
      FPX = 0.0;
      FPY = 0.0;
      FPZ = 0.0;
      for I = 1:4,
         FPX = FPX+XTHRST(I);
         FPY = FPY+YTHRST(I);
         FPZ = FPZ+ZTHRST(I);
      end

%%% COMPUTE ROTATIONAL EFFECTS (TORQUE) DUE TO ENGINE OFFSET %%%
%%% FROM CENTERLINE, AND THRUST VECTORING                          %%%
TORQUE = zeros(3,1);
      for I = 1:4,
         TVECLC(I,1) = TLOCAT(I,1)-DXTHRS(I)*CSANXZ(I)*CSANXY(I);
         TVECLC(I,2) = TLOCAT(I,2)-DXTHRS(I)*CSANXZ(I)*SNANXY(I);
         TVECLC(I,3) = TLOCAT(I,3)+DXTHRS(I)*SNANXZ(I);
         TORQUE(1) = TORQUE(1)+ZTHRST(I)*TVECLC(I,2)-...
                     YTHRST(I)*TVECLC(I,3);
         TORQUE(2) = TORQUE(2)+XTHRST(I)*TVECLC(I,3)-...
                     ZTHRST(I)*TVECLC(I,1);
         TORQUE(3) = TORQUE(3)+YTHRST(I)*TVECLC(I,1)-...
                     XTHRST(I)*TVECLC(I,2);
      end

%%% COMPUTE ANGULAR MOMENTUM OF ENGINES %%%
      for I = 1:4,
         OMEGX =  ENGOMG(I)*CSANXZ(I)*CSANXY(I);
         OMEGY =  ENGOMG(I)*CSANXZ(I)*SNANXY(I);
         OMEGZ = -ENGOMG(I)*SNANXZ(I);
         EIXE  = EIX(I);
         EX    = TLOCAT(I,1);
         EY    = TLOCAT(I,2);
         EZ    = TLOCAT(I,3);
     %%% COMPUTE THE INERTIA TENSOR OF THE I-TH ENGINE  %%%
     %%% ABOUT THE AIRCRAFT REFERENCE C.G.              %%%
         AINRTA(1,1) =   EIXE*CSANXZ(I)*CSANXZ(I)*...
                         CSANXY(I)*CSANXY(I);
         AINRTA(1,2) =   EIXE*CSANXZ(I)*CSANXZ(I)*...
                         SNANXY(I)*CSANXY(I);
         AINRTA(1,3) =  -EIXE*CSANXZ(I)*SNANXZ(I)*...
                         CSANXY(I);
         AINRTA(2,1) =   AINRTA(1,2);
         AINRTA(2,2) =   EIXE*SNANXY(I)*SNANXY(I)*...
```

```
                         CSANXZ(I)*CSANXZ(I);
        AINRTA(2,3) =   -EIXE*SNANXZ(I)*CSANXZ(I)*SNANXY(I);
        AINRTA(3,1) =    AINRTA(1,3);
        AINRTA(3,2) =    AINRTA(2,3);
        AINRTA(3,3) =    EIXE*SNANXZ(I)*SNANXZ(I);
        for J = 1:3,
           HENGIN(I,J) =  OMEGX*AINRTA(J,1)+OMEGY*...
                          AINRTA(J,2)+OMEGZ*AINRTA(J,3);
        end
     end

%%% COMPUTE GYROSCOPIC EFFECTS %%%
GYRO = zeros(3,1);
      for I = 1:4,
         GYRO(1) = GYRO(1)+Q*HENGIN(I,3)-R*HENGIN(I,2);
         GYRO(2) = GYRO(2)+R*HENGIN(I,1)-P*HENGIN(I,3);
         GYRO(3) = GYRO(3)+P*HENGIN(I,2)-Q*HENGIN(I,1);
      end

%%% COMPUTE TOTAL MOMENT INCREMENT DUE TO ENGINES %%%
      DCL   = GYRO(1)+TORQUE(1);
      DCM   = GYRO(2)+TORQUE(2);
      DCN   = GYRO(3)+TORQUE(3);
```

- MATLAB function: uengin

```
function [TAUL,TAUR,PLAL,PLAR,COUT1C,COUT2C,FIRST] = uengin(A,IA);

%%% ROUTINE TO MODEL THE ENGINE %%%
      FIRST = IA(502);   %%% Logical TRUE %%%

      %%% COMMON BLOCK
      global THRUST TLOCAT XYANGL XZANGL TVANXY TVANXZ DXTHRS
      global EIX AMSENG ENGOMG
      if FIRST==1
         FIRST = 0;    %%% Logical FALSE %%%
         THRUST = zeros(4,1);
         TLOCAT = zeros(4,3);
         XYANGL = zeros(4,1);
         XZANGL = zeros(4,1);
         TVANXY = zeros(4,1);  %%% ADDED, DUTTON 1/20/94
         TVANXZ = zeros(4,1);  %%% ADDED, DUTTON 1/20/94
         DXTHRS = zeros(4,1);  %%% ADDED, DUTTON 1/20/94
         ENGOMG = zeros(4,1);  %%% ADDED, DUTTON 1/20/94
         EIX    = zeros(4,1);  %%% ADDED, DUTTON 1/20/94
%%% ENGINES: 2 ENGINES 10 FT. BEHIND CG, 4 FT OFF CENTERLINE %%%
         TLOCAT(1,1) = -10.0;
         TLOCAT(1,2) = -4.0;
         TLOCAT(2,1) = -10.0;
         TLOCAT(2,2) =  4.0;
      end

%%% CALL ROUTINE TO MODEL ENGINE %%%
[TAUL,TAUR,PLAL,PLAR,COUT1C,COUT2C] = engmdl(A,IA);
```

- MATLAB function: engmdl

```
function [TAUL,TAUR,PLAL,PLAR,COUT1C,COUT2C] = engmdl(A,IA);

%%% ROUTINE TO IMPLEMENT FIRST ORDER ENGINE MODEL %%%
%%% WRITTEN AUG. 2, 1978 %%%
%%%   LEE DUKE NASA/DFRC %%%

%%% COMMON BLOCKS
      %%% ENGVAR
global C03RTN C03RTP C05F1K C06LMN C06LMP B02LMN B02LMP
global B03RTN B03RTP B04F1K B06F1E B06F1K
```

```
      %%% TLUENG
global TIDL TMIL TMIN TMAX FIDL FMIL FMIN FMAX
      %%% CONPOS
global DAP FLAT DATRIM DEP FLON DETRIM DRP FPED DRTRIM DSBP
global DFP THSL THSR
      %%% ENGSTF
global THRUST TLOCAT XYANGL XZANGL TVANXY TVANXZ DXTHRS
global EIX AMSENG ENGOMG

%%% ASSIGN A,IA-ARRAY VAR NAMES %%%
IMODE  = IA(501);  PLAPL  = A(1416);
PLAPR  = A(1417); PLASYM = A(1418);


%%% ADDED, DUTTON 4/7/94 %%%
COUT1C = A(667);   COUT2C = A(668);

%%% ASSIGN DATA VALUES %%%
CORMIN = 20.0; CORDIS = 63.0;
AUGMIN = 83.1; AUGDIS = 44.0;
FMSSIC = 0.0;
B04IN2 = 0.0; B04OT2 = 0.0;
B06IN2 = 0.0; B06OT2 = 0.0;
B14IN2 = 0.0; B14OT2 = 0.0;
B16IN2 = 0.0; B16OT2 = 0.0;
C05IN2 = 0.0; C05OT2 = 0.0;
C06OT2 = 0.0; C07OT2 = 0.0;
C15IN2 = 0.0; C15OT2 = 0.0;
C16OT2 = 0.0; C17OT2 = 0.0;
      ENGLU = 1.02;
      ENGRU = .97;
      if PLASYM > 1e-10
        PLAPL = PLASYM;
        PLAPR = PLASYM;
        ENGLU = 1.0;
        ENGRU = 1.0;
      end

%%% FIND THRUST AND FUEL FLOW FOR PRESENT %%%
%%% MACH/ALTITUDE CONDITION %%%
H    = A(713);
AMCH = A(825);
engtlu(H,AMCH);

%%% ENGINE COMMAND INPUTS %%%
%%% RIGHT ENGINE CALCULATIONS %%%
%%% DETERMINE INPUTS TO CORE AND AUGMENTOR MODELS %%%
      CIN01 = (PLAPR-CORMIN)/CORDIS;
      [CIN01] = flimit(CIN01, 0.0, 1.0);
      BIN01 = (PLAPR-AUGMIN)/AUGDIS;
      [BIN01] = flimit(BIN01, 0.0, 1.0);

%%% WAIT FOR CORE TO SPOOL UP TO 99% BEFORE BURNER IS ACTIVE %%%
      if COUT1C < 0.99
        BIN01 = 0.0;
        B04OT2 = 0.0;
        B05TIM = 0.0;
      end

%%% FIRST ORDER CORE ENGINE MODEL %%%
      if IMODE==5
        C02JN1 = CIN01-C06OT2;
        C03IN1 = C02JN1;
        [C03OT1] = flimit(C03IN1,C03RTN,C03RTP);
        C04IN1 = C03OT1;
        C04MLT = C07OT2;
        C04OT1 = C04IN1*C04MLT;
        C05IN1 = C04OT1;
        C05OT1 = C05OT2+C05F1K*(C05IN1+C05IN2);
      else
        C05OT1 = CIN01;
      end
      C06IN1 = C05OT1;
      [C06OT1] = flimit(C06IN1,C06LMN,C06LMP);
      C07IN1 = C05OT1;
      [C07OT1] = c07sdl(C07IN1);
      if abs(C07OT2) >= 1e-10
        TAUR = 1/C07OT2;
      else
        TAUR = 1e10;
      end
      COUT1C = C06OT1;

%%% SIMPLE AUGMENTOR MODEL %%%
      if IMODE==5
        B01JN1 = BIN01-B05OT2;
        B02IN1 = B01JN1;
        B02OT1 = B02IN1;
        B03IN1 = B02OT1;
        [B03OT1] = flimit(B03IN1,B03RTN,B03RTP);
        B04IN1 = B03OT1;
        B04OT1 = B04OT2+B04F1K*(B04IN1+B04IN2);
      else
        B04OT1 = BIN01;
      end
      [B04OT1] = flimit(B04OT1,0.0,1.0);
      B05OT1 = B04OT1;
      B06IN1 = B05OT1;
      if IMODE==5
```

```
    B06OT1 = B06OT2*B06F1E+B06F1K*(B06IN1+B06IN2);
  else
    B06OT1 = B05OT1;
  end
  B07IN1 = B06OT1;
  [B07OT1] = b07sdl(B07IN1);
  B08IN1 = B07OT1;
  [B08OT1] = b08fcn(B08IN1);
  BOUT1C = B08OT1;
  BOUT2C = B07OT1;

%%% OUTPUTS %%%
  COROTR = COUT1C;
  AUGOTR = BOUT1C;
  AUGPLR = BOUT2C;

%%% UPDATE FOR NEXT FRAME %%%
  C05OT2 = C05OT1;
  C05IN2 = C05IN1;
  C06OT2 = C06OT1;
  C07OT2 = C07OT1;
  B04OT2 = B04OT1;
  B04IN2 = B04IN1;
  B05OT2 = B05OT1;
  B06OT2 = B06OT1;
  B06IN2 = B06IN1;

%%% LEFT ENGINE CALCULATIONS %%%
%%% DETERMINE INPUTS TO CORE AND AUGMENTOR MODELS %%%
  CIN11 = (PLAPL-CORMIN)/CORDIS;
  [CIN11] = flimit(CIN11,0.0,1.0);
  BIN11 = (PLAPL-AUGMIN)/AUGDIS;
  [BIN11] = flimit(BIN11,0.0,1.0);
```

```
%%% CORE TO AUGMENTOR SWITCHING LOGIC %%%
%%% WAIT FOR CORE TO SPOOL UP TO 99% BEFORE BURNER IS ACTIVE %%%
      if COUT2C < 0.99
        BIN11 = 0.0;
        B14OT2 = 0.0;
        B15TIM = 0.0;
      end

%%% FIRST ORDER CORE ENGINE MODEL %%%
      if IMODE==5
        C12JN1 = CIN11-C16OT2;
        C13IN1 = C12JN1;
        [C13OT1] = flimit(C13IN1,C03RTN,C03RTP);
        C14IN1 = C13OT1;
        C14MLT = C17OT2;
        C14OT1 = C14IN1*C14MLT;
        C15IN1 = C14OT1;
        C15OT1 = C15OT2+C05F1K*(C15IN1+C15IN2);
      else
        C15OT1 = CIN11;
      end
      C16IN1 = C15OT1;
      [C16OT1] = flimit(C16IN1,C06LMN,C06LMP);
      C17IN1 = C15OT1;
      [C17OT1] = c07sdl(C17IN1);
      if abs(C17OT2) >= 1e-10
        TAUL = 1/C17OT2;
      else
        TAUL = 1e10;
      end
      COUT2C = C16OT1;
```

```
%%% SIMPLE AUGMENTOR MODEL %%%
   if IMODE==5
    B11JN1 = BIN11-B15OT2;
    B12IN1 = B11JN1;
    B12OT1 = B12IN1;
    B13IN1 = B12OT1;
    [B13OT1] = flimit(B13IN1,B03RTN,B03RTP);
    B14IN1 = B13OT1;
    B14OT1 = B14OT2+B04F1K*(B14IN1+B14IN2);
   else
    B14OT1 = BIN11;
   end
   [B14OT1] = flimit(B14OT1,0.0,1.0);
   B15OT1 = B14OT1;
   B16IN1 = B15OT1;
   if IMODE==5
    B16OT1 = B16OT2*B06F1E+B06F1K*(B16IN1+B16IN2);
   else
    B16OT1 = B15OT1;
   end
   B17IN1 = B16OT1;
   [B17OT1] = b07sdl(B17IN1);
   B18IN1 = B17OT1;
   [B18OT1] = b08fcn(B18IN1);
   BOUT3C = B18OT1;
   BOUT4C = B17OT1;

%%% OUTPUTS %%%
   COROTL = COUT2C;
   AUGOTL = BOUT3C;
   AUGPLL = BOUT4C;

%%% UPDATE FOR NEXT FRAME %%%
```

```
      C15OT2 = C15OT1;
      C15IN2 = C15IN1;
      C16OT2 = C16OT1;
      C17OT2 = C17OT1;
      B14OT2 = B14OT1;
      B14IN2 = B14IN1;
      B15OT2 = B15OT1;
      B16OT2 = B16OT1;
      B16IN2 = B16IN1;

%%% COMPUTE ENGINE MODEL OUTPUTS %%%
      if COROTR < 0.99
        PLAR = COROTR*CORDIS+CORMIN;
        FLOWR = (FMIL-FIDL)*COROTR+FIDL;
        THSR = (TMIL-TIDL)*COROTR+TIDL;
      else
        PLAR = COROTR*CORDIS+AUGOTR*AUGDIS+CORMIN;
        FLOWR = (FMAX-FMIL)*AUGOTR+(FMIL-FIDL)*COROTR+FIDL;
        THSR = (TMAX-TMIL)*AUGOTR+(TMIL-TIDL)*COROTR+TIDL;
      end
      if COROTR < 0.99
        PLAL = COROTL*CORDIS+CORMIN;
        FLOWL = (FMIL-FIDL)*COROTL+FIDL;
        THSL = (TMIL-TIDL)*COROTL+TIDL;
      else
        PLAL = COROTL*CORDIS+AUGOTL*AUGDIS+CORMIN;
        FLOWL = (FMAX-FMIL)*AUGOTL+(FMIL-FIDL)*COROTL+FIDL;
        THSL = (TMAX-TMIL)*AUGOTL+(TMIL-TIDL)*COROTL+TIDL;
      end

%%% STORE IN COMMON %%%
      THRUST(1) = ENGRU*THSR;
      THRUST(2) = ENGLU*THSL;
```

E.4 Atmospheric Model Listing

The following is a listing of the modules which comprise the atmospheric model used in the nonlinear F-15 simulation.

- MATLAB function: atmos

```
function [AMCH,RHO,QBAR,G] = atmos(A);

%%% ROUTINE TO CALCULATE AIR DATA PARAMETERS %%%
%%% WRITTEN MAY 13, 1981 %%%
%%%   LEE DUKE NASA/DFRC %%%

%%% ASSIGN A-ARRAY VAR NAMES %%%
V = A(829);
H = A(713);

%%% INVOKE ATMOSPHERIC MODEL %%%
   [An,RHO,G,PA,TMPR,VMU] = altfn(H);

%%% COMPUTE MACH %%%
   AMCH = V/An;

%%% COMPUTE DYNAMIC PRESSURE %%%
   QBAR = RHO*V^2/2.0;
```

- MATLAB function: altfn

```
function [A,RHO,G,PA,TMPR,VMU] = altfn(H);

%%%   ROUTINE PROVIDES BY TABLE LOOK UP                    %%%
%%%         VELOCITY OF SOUND            A - (FT/SEC)      %%%
%%%         ACCELERATION DUE TO GRAVITY  G - (FT/SEC**2)   %%%
%%%         AIR DENSITY                RHO - (SLUGS/FT**3) %%%
%%%         AMBIENT STATIC PRESSURE    PA - (PSF)          %%%
%%%         AMBIENT AIR TEMPERATURE  TMPR - (DEG RANKIN)  %%%

%%% AS A FUNCTION OF ALTITUDE, H (FT)
%%% BREAK POINTS ON H ARE 3280.84 FT (1000.0 M)
%%% The data array is of size 81.
%%% The first data XX(1) corresponds to SEA LEVEL to 999m
%%% The second data XX(2) corresponds to 1000m to 1999m etc...
%%% Equal to or above 80,000m the last array element is XX(81).

%%% WRITTEN 11/9/72 A MYERS  NASA FRC
%%% TMPR ADDED   1 MAY 78   L SCHILLING   NASA/DFRC

AA = [1116.45,1103.79,1090.98,1078.03,1064.92,1051.66,1038.23...
      1024.63,1010.84, 996.88, 982.72, 968.07, 968.07, 968.07...
      968.07, 968.07, 968.07, 968.07, 968.07, 968.07, 968.07...
      970.15, 972.37, 974.57, 976.77, 978.97, 981.15, 983.34...
      985.52, 987.69, 989.86, 992.02, 994.28, 999.56,1005.54...
      1011.48,1017.39,1023.25,1029.08,1034.88,1040.65,1046.38...
      1052.07,1057.74,1063.37,1068.97,1074.54,1082.02,1082.02...
      1082.02,1082.02,1082.02,1082.02,1079.77,1075.82,1071.86...
      1067.89,1063.90,1059.90,1055.89,1051.86,1047.81,1042.09...
      1033.92,1025.68,1017.39,1009.02,1000.59, 992.08, 983.51...
      974.87, 966.15, 957.35, 948.47, 939.52, 930.48, 921.36...
      912.14, 902.82, 893.44, 883.99];
RHOA = [.23769D-2,.21571D-2,.19531D-2,.17642D-2,.15898D-2...
   .14289D-2,.12808D-2,.11448D-2,.10202D-2,.90625D-3,.80234D-3...
   .70783D-3,.60526D-3,.51729D-3,.44212D-3,.37788D-3,.32301D-3...
   .27611D-3,.23604D-3,.20179D-3,.17251D-3,.14691D-3,.12517D-3...
   .10673D-3,.91075D-4,.77776D-4,.66470D-4,.56848D-4,.48655D-4...
   .41674D-4,.35721D-4,.30642D-4,.26301D-4,.22455D-4,.19185D-4...
   .16422D-4,.14083D-4,.12099D-4,.10413D-4,.89773D-5,.77529D-5...
   .67065D-5,.58109D-5,.50427D-5,.43830D-5,.38153D-5,.33259D-5...
```

```
    .29037D-5,.25548D-5,.22562D-5,.19925D-5,.17597D-5,.15541D-5...
    .13782D-5,.12251D-5,.10880D-5,.96554D-6,.85609D-6,.75839D-6...
    .67123D-6,.59358D-6,.52443D-6,.46434D-6,.41236D-6,.36554D-6...
    .32335D-6,.28548D-6,.25154D-6,.22118D-6,.19403D-6,.16985D-6...
    .14823D-6,.12921D-6,.11228D-6,.97309D-7,.84113D-7,.72490D-7...
    .62284D-7,.53359D-7,.45578D-7,.38787D-7];
GA = [32.174,32.164,32.154,32.144,32.134,32.123,32.114,32.103...
      32.093,32.083,32.073,32.063,32.053,32.043,32.033,32.023...
      32.013,32.003,31.992,31.983,31.972,31.963,31.952,31.943...
      31.932,31.923,31.912,31.903,31.892,31.883,31.872,31.863...
      31.852,31.843,31.833,31.823,31.813,31.803,31.793,31.783...
      31.773,31.763,31.753,31.743,31.733,31.723,31.713,31.703...
      31.694,31.684,31.674,31.664,31.654,31.644,31.634,31.624...
      31.615,31.605,31.595,31.585,31.575,31.565,31.555,31.546...
      31.536,31.526,31.516,31.506,31.496,31.487,31.477,31.467...
      31.457,31.448,31.438,31.427,31.417,31.407,31.398,...
      31.388,31.378];
PAA = [.211622D+4,.187711D+4,.166042D+4,.146451D+4,.128781D+4...
       .112882D+4,.986161D+3,.858503D+3,.744600D+3,.643286D+3...
       .553461D+3,.474097D+3,.405167D+3,.346272D+3,.295953D+3...
       .252961D+3,.216223D+3,.184830D+3,.158003D+3,.135076D+3...
       .115482D+3,.987659D+2,.845338D+2,.724070D+2,.620662D+2...
       .532416D+2,.457051D+2,.392640D+2,.337551D+2,.290396D+2...
       .250005D+2,.215383D+2,.185685D+2,.160256D+2,.138557D+2...
       .120006D+2,.104118D+2,.904857D+1,.787671D+1,.686758D+1...
       .599712D+1,.524502D+1,.459412D+1,.402987D+1,.354002D+1...
       .311404D+1,.274311D+1,.241960D+1,.213649D+1,.188672D+1...
       .166622D+1,.147155D+1,.129967D+1,.114774D+1,.101277D+1...
       .892889D+0,.786488D+0,.692138D+0,.608547D+0,.534546D+0...
       .469102D+0,.411271D+0,.360183D+0,.314857D+0,.274653D+0...
       .239062D+0,.207616D+0,.179888D+0,.155489D+0,.134066D+0...
       .115298D+0,.988932D-1,.845889D-1,.721470D-1,.613526D-1...
       .520117D-1,.439533D-1,.370199D-1,.310722D-1,...
```

```
         .259855D-1,.216500D-1];
TMPRA = [518.670, 506.972, 495.277, 483.586, 471.899...
          460.217, 448.537, 436.860, 425.187, 413.159...
          401.854, 390.193, 389.970, 389.970, 389.970...
          389.970, 389.970, 389.970, 389.970, 389.970...
          389.970, 391.646, 393.433, 395.221, 397.008...
          398.794, 400.579, 402.365, 404.149, 405.932...
          407.716, 409.500, 411.282, 415.751, 420.737...
          425.723, 430.708, 435.690, 440.672, 445.651...
          450.630, 455.605, 460.580, 465.554, 470.525...
          475.495, 480.465, 485.431, 487.170, 487.170...
          487.170, 487.170, 487.170, 485.149, 481.608...
          478.069, 474.530, 470.993, 467.458, 463.923...
          460.390, 456.858, 451.883, 444.821, 437.764...
          430.708, 423.653, 416.603, 409.552, 402.505...
          395.460, 388.418, 381.377, 374.339, 367.303...
          360.270, 353.232, 346.212, 339.174, 332.154, 325.170];

%%% DEFINE THE BREAK POINT
FTBP = 3280.84;
%%% INTERPOLATION OF THE DATA
   if H <= 0.0
      A    = AA(1);
      RHO  = RHOA(1);
      G    = GA(1);
      PA   = PAA(1);
      TMPR = TMPRA(1);
%%% EQUATION FOR VISCOSITY (VMU) ACCURATE TO 90 KM ONLY %%%
      VMU = ((7.3025D-07)*(TMPR)^(1.5))/(TMPR+198.72);
   elseif H >= 262467.2
      A    = AA(81);
      RHO  = RHOA(81);
      G    = GA(81);
      PA   = PAA(81);
      TMPR = TMPRA(81);
%%% VISCOSITY ABOVE 90 KM SET AT 90 KM VALUE %%%
      VMU = 8.32627D-06;
   else
      RBP  = H/FTBP;
      I1   = floor(RBP)+1;
      I2   = I1+1;
      R    = RBP-I1+1;
      A    = R*(AA(I2)-AA(I1))+AA(I1);
      RHO  = R*(RHOA(I2)-RHOA(I1))+RHOA(I1);
      G    = R*(GA(I2)-GA(I1))+GA(I1);
      PA   = R*(PAA(I2)-PAA(I1))+PAA(I1);
      TMPR = R*(TMPRA(I2)-TMPRA(I1))+TMPRA(I1);
%%% EQUATION FOR VISCOSITY (VMU) IS ACCURATE UP TO 90 KM ONLY
      VMU = ((7.3025D-07)*(TMPR)^(1.5))/(TMPR+198.72);
   end
```

Appendix F
SAMPLE SIMULATION COMMAND LISTING

F.1 Nonlinear Open-Loop Simulation

The following summarizes the flow of commands to execute the nonlinear open-loop F-15 simulation:

- Start MATLAB.
- Using an editor, enter the desired trim values for x_o and u_o in `trimmod.m`.
- Type `f25load` in MATLAB.
- Type `f25sim` in MATLAB to load the SIMULINK environment.
- Select *Simulation* menu, select *Parameters*, and set desired simulation parameters. Note: The open-loop simulations in this report used the Euler method of numerical integration.
- Select *Simulation* menu and select *Start.*
- If you desire to monitor simulation's progress, double-click on the Clock icon and a running count of the time will be shown.
- Once the simulation is complete, the selected parameters are stored in the MATLAB workspace.

F.2 Nonlinear Closed-Loop Simulation

The following summarizes the flow of commands to execute the nonlinear closed-loop F-15 simulation:

- Start MATLAB.

- Using an editor, enter the desired trim values for x_o and u_o in `trimmod.m`.

- Type `f25load` in MATLAB.

- Enter the values for the TECS controller gains. If either FP1 or FP2 is being used, these gains can be loaded by typing `load f25sd539_cx` or `load f25sd497_cx` respectively.

- Type `f25simtecs` in MATLAB to load the SIMULINK environment.

- Select *Simulation* menu, select *Parameters*, and set desired simulation parameters. Note: The closed-loop simulations in this report used the Euler method of numerical integration and minimum time steps of no less than 0.01.

- Select *Simulation* menu and select *Start.*

- If you desire to monitor simulation's progress, double-click on the Clock icon and a running count of the time will be shown.

- Once the simulation is complete, the selected parameters are stored in the MATLAB workspace.

- To quickly view the output, type `plotcl` in MATLAB.

F.3 Linear Closed-Loop Simulation

The following summarizes the flow of commands to execute the linearized closed-loop F-15 simulation:

- Start MATLAB.

- Load the state space model. If either FP1 or FP2 is being used, they can be loaded by typing `trim497ss` or `trim539ss` respectively.

- Enter the values for the TECS controller gains. If either FP1 or FP2 is being used, these gains can be loaded by typing `load f25sd539_cx` or `load f25sd497_cx` respectively.

- Type `f25simtecslin` in MATLAB to load the SIMULINK environment.

- Select *Simulation* menu, select *Parameters*, and set desired simulation parameters. Note: The closed-loop simulations in this report used the Euler method of numerical integration and minimum time steps of no less than 0.01.

- Select *Simulation* menu and select *Start.*

- If you desire to monitor simulation's progress, double-click on the Clock icon and a running count of the time will be shown.

- Once the simulation is complete, the selected parameters are stored in the MATLAB workspace.

- To quickly view the output, type `plotlin` in MATLAB.